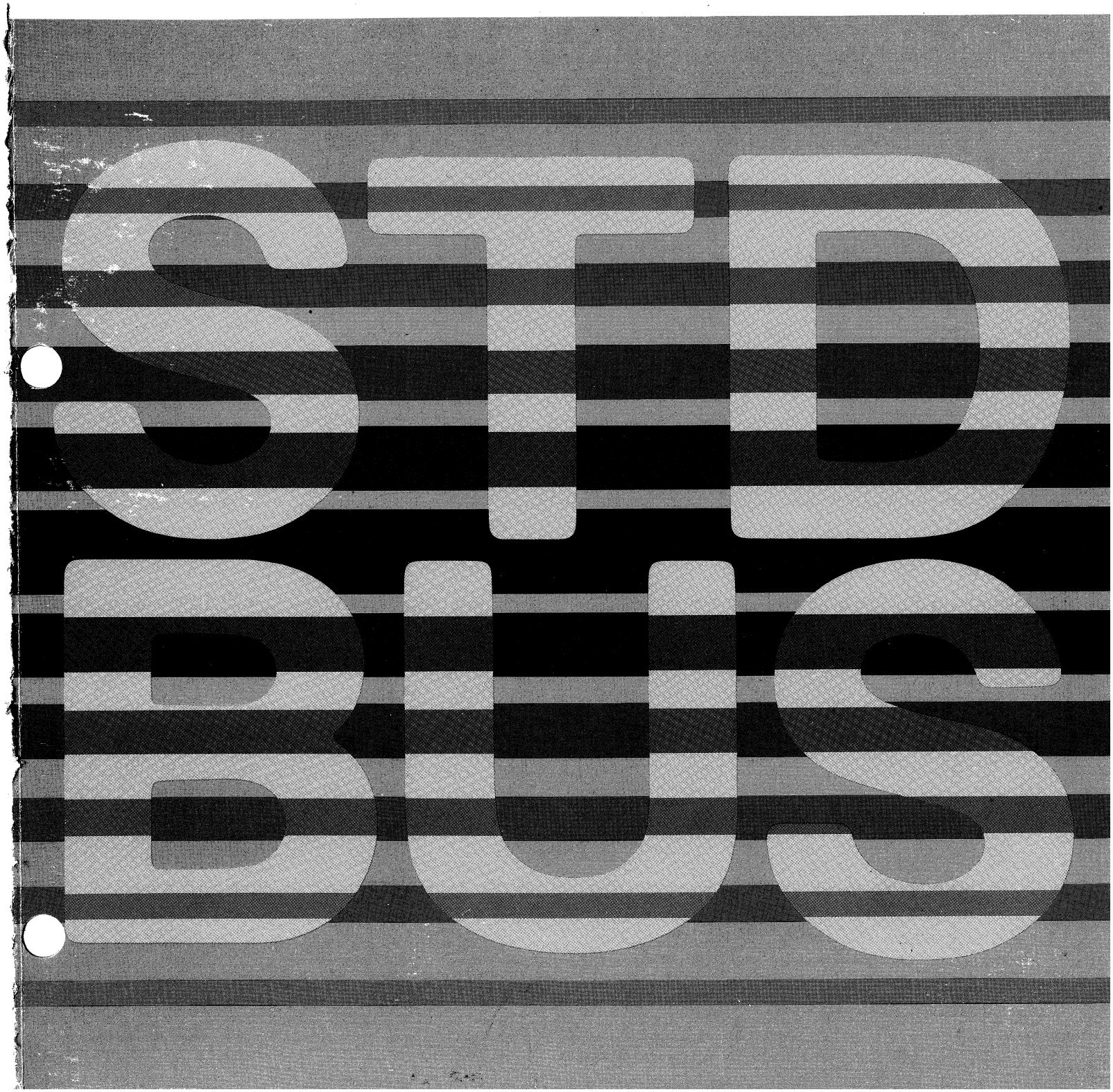
PRO-LOG®
CORPORATION

STD 7000

7314A/7312 Serial Communications Controller Card

USER'S MANUAL

A Pro-Log OPENWARE™ Product



NOTICE

The information in this document is provided for reference only. Pro-Log does not assume any liability arising out of the application or use of the information or products described herein.

This document may contain or reference information and products protected by copyrights or patents and does not convey any license under the patent rights of Pro-Log, nor the rights of others.

Printed in U.S.A. Copyright 1986 by Pro-Log Corporation, Monterey, CA 93940. All rights reserved. However, any part of this document **may be reproduced** with Pro-Log Corporation cited as the source.

PRO-LOG®
CORPORATION

STD 7000

7314A/7312 Serial Communications Controller Card

USER'S MANUAL

A Pro-Log OPENWARE™ Product

2/86

FOREWORD

This manual explains how to use Pro-Log's 7314A and 7312 cards. It is structured to reflect the answers to basic questions about the 7314A and 7312. We welcome your suggestions on how we can improve our instructions.

The 7314A and 7312 are OPENWARE products in the Pro-Log Series 7000 STD BUS hardware. Our products are modular and provide industrial managers with the means of using their own people to control the design, production, and maintenance of products that use STD BUS hardware. Pro-Log also offers applications seminars based on STD BUS products.

OPENWARE, a trademark of Pro-Log Corporation, means our products are designed to quality standards with readily available, second-sourced, industry-standard components, clear and complete specifications, full documentation, and the option to purchase the manufacturing rights and drawings.

You may also find the *STD BUS Technical Manual and Data Book* useful in your work. If you would like a copy of this document, please write to us on your company letterhead.

Contents

	Page
Foreword	ii
Figures	iv
References & Bibliography	v
Section 1. Specifications	1-1
Purpose	1-1
Features	1-1
Functional Capability	1-2
Interrupt Prioritization	1-3
I/O Port Mapping	1-3
Specifications	1-4
Electrical	1-4
Timing	1-4
Mechanical	1-11
Reference Specifications	1-11
Section 2. Installation	2-1
Introduction	2-1
Installation	2-1
Card Layout and Jumpers	2-2
I/O Address Mapping	2-4
8-bit Port Addressing Systems	2-4
10-bit Port Addressing Systems	2-5
Interrupt Vector Enable	2-8
Interrupt Request	2-8
Priority Chain	2-8
Select Input	2-8
RS-232-C Interface	2-8
Card Configuration	2-9
Section 3. Operation and Programming	3-1
Introduction	3-1
Serial Communications Capabilities	3-1
SCC Internal Structure	3-2
SCC Receive and Transmit Registers	3-7
SCC Control Registers	3-7
Write Register 0	3-8
Write Register 1	3-9
Write Register 2	3-10
Write Register 3	3-10
Write Register 4	3-11
Write Register 5	3-11
Write Register 6	3-12
Write Register 7	3-12
Write Register 8	3-12
Write Register 9	3-12
Write Register 10	3-13
Write Register 11	3-14
Write Register 12	3-14
Write Register 13	3-14
Write Register 14	3-15
Write Register 15	3-15
SCC Status Registers	3-17
Read Register 0	3-17
Read Register 1	3-18
Read Register 2	3-19
Read Register 3	3-19
Read Register 8	3-20
Read Register 10	3-20
Read Register 12	3-20
Read Register 13	3-20
Read Register 15	3-20
SCC Operation	3-21
Initialization	3-21
Baud Rates	3-22
Interrupts	3-23
RS-232-C Signals	3-24
Section 4. Operating Software	4-1
Introduction	4-1
Memory Addresses	4-1
I/O Port Addresses	4-1
Description	4-1
Subroutine Specifications	4-2
(INIT ASYN COMM)	4-2
(INIT SCC X)	4-2
(INIT SCC CH)	4-3
(XMT CH A ASYNC)	4-3
(RCV CH B ASYNC)	4-3
(INIT SDLC COMM)	4-3
(INIT INTER)	4-3
(XMT CH C SDLC)	4-4
(XMT BLK C)	4-4
(RCV CH D SDLC)	4-4
(SPCL RCVD SDLC)	4-4
Section 5. Maintenance	5-1
Reference Drawings	5-1
Return for Repair/Warranty	5-6

Figures

Figure		Page
1-1	7314A SCC Card	1-1
1-2	7312 SCC Card	1-1
1-3	Block Diagram	1-2
1-4	RS-232-C	1-2
1-5	Interrupt Request Connector J1	1-3
1-6	SCC Interrupt Source Priority	1-3
1-7	I/O Port Addressing	1-3
1-8	7314A-1/7312-1 Input Timing With an 8088 CPU	1-5
1-9	7314A-1/7312-1 Output Timing With an 8088 CPU (advanced write)	1-5
1-10	7314A-1/7312-1 Output Timing With an 8088 CPU (normal write)	1-6
1-11	7314A-1/7312-1 Interrupt Acknowledge Timing with an 8088 CPU	1-6
1-12	7314A-2/7312-2 Input Timing with a Z80 CPU	1-7
1-13	7314A-2/7312-2 Output Timing with a Z80 CPU	1-7
1-14	7314A-2/7312-2 Interrupt Acknowledge Timing With a Z80 CPU	1-8
1-15	7314A-2/7312-2 Input Timing with an 8085 CPU	1-9
1-16	7314A-2/7312-2 Output Timing with an 8085 CPU	1-9
1-17	7314A-2/7312-2 Interrupt Acknowledge Timing with an 8085 CPU	1-10
1-18	STD Edge Connector	1-11
1-19	Cables and Connectors	1-11
2-1	Installation	2-1
2-2	Jumper Configuration	2-2
2-3	Sources for Shorting Jumpers	2-2
2-4	Physical Layout and Jumper Options	2-2
2-5	Eight-Bit Port Address Mapping	2-4
2-6	1K I/O Port Address Map	2-5
2-7	Ten-Bit Port Address Mapping	2-6
2-8	RS-232-C Serial Ports DCE/DTE Jumpers	2-9
2-9	RS-232-C Serial Port D Connector Converter	2-9

Figure		Page
3-1	Examples of Various Serial Communications Formats	3-1
3-2	SCC Register Structure	3-2
3-3	Write Register Bit Maps	3-3
3-4	Read Register Bit Maps	3-6
3-5	SCC Control Register Functions	3-8
3-6	Read Register Functions	3-17
3-7	Status Vector Mode Table	3-19
3-8	SCC Initialization Order	3-21
3-9	Time Constants for Common Baud Rates	3-22
3-10	Maximum T-States for Simultaneous Operation	3-22
3-11	Vectored Interrupt Priority	3-23
3-12	Signal Flow in DCE Configuration	3-25
3-13	Signal Flow in DTE Configuration	3-25
3-14	SCC RS-232-C Handshake Capability	3-26
4-1	Subroutine Types	4-1
4-2	Main Subroutine Use Requirements	4-2
4-3	1K Program Memory Map	4-5
4-4	(INIT ASYN COMM) Flowchart	4-6
4-5	(INIT SCC X) Flowchart	4-6
4-6	(INIT SCC CH) Flowchart	4-6
4-7	(XMT CH A ASYNC) Flowchart	4-7
4-8	(RCV CH B ASYNC) Flowchart	4-8
4-9	(INIT SDLC COMM) Flowchart	4-8
4-10	(INIT INTER) Flowchart	4-9
4-11	(XMT CH C SDLC) Flowchart	4-9
4-12	(XMT BLK C) Flowchart	4-10
4-13	(RCV CH D SDLC) Flowchart	4-11
4-14	(SPCL RCV D SDLC) Flowchart	4-11
4-15	(ASYNC COMM) Flowchart	4-12
4-16	(SDLC COMM) Flowchart	4-12
4-17	Program Listing (Z80)	4-13
4-18	Program Listing (8088)	4-26
5-1	Schematic	5-2
5-2	Assembly Drawing	5-5

REFERENCES/BIBLIOGRAPHY

Z8030/Z8530 SCC Serial Communications Controller Technical Manual, Zilog, Inc., 1983.

AMZ8030/AMZ8530 Serial Communications Controller Technical Manual, Advance Micro Devices Inc., 1982.

82530 SCC Technical Manual, Intel Corporation, 1984.

*EIA Standard RS-232-C**

Overview of Serial Data Communications. Pro-Log STD BUS Application Note #156, Pro-Log Corporation, 1985. (A copy of this Application Note is provided with this manual.)

*To obtain copies of EIA Standards, write to:

Electronic Industries Association Engineering Department,
2001 Eye Street, N.W. Washington, D.C., 20006

Section 1
SPECIFICATIONS

PURPOSE

The 7314A and 7312 provide fully independent, full-duplex RS-232-C serial channels that can be used to expand the communications interface for 8088, Z80, and 8085 STD BUS systems. The 7314A has four serial channels and the 7312 has two serial channels.

The 7314A/7312 easily interface the STD BUS to equipment with RS-232-C channels, such as: terminals, printers, modems, displays, robotic arms, instrumentation, PROM Programmers, mainframes, minicomputers, and other microcomputers. The 73124A/7312 are found in a wide variety of applications such as distributed processing, data concentration, development systems, process control, and automatic test equipment.

FEATURES

- Four (7314A) or two (7312) Independent, full-duplex RS-232-C serial communications channels
- 8088 (5 MHz) compatible (7314A-1, 7312-1)
- Z80 (2.5, 4.0 MHz), and 8085A (6.25 MHz) compatible (7314A-2, 7312-2)
- Asynchronous and synchronous protocols (BISCYNC, SDLC, HDLC, and CCITT-X.25)
- Programmable baud rates to greater than 19.2K baud
- Generates prioritized vectored interrupts from external or on-card sources
- Communications diagnostics can be easily designed using Local Loopback mode
- Configurable for DTE or DCE applications
- Two-year limited warranty

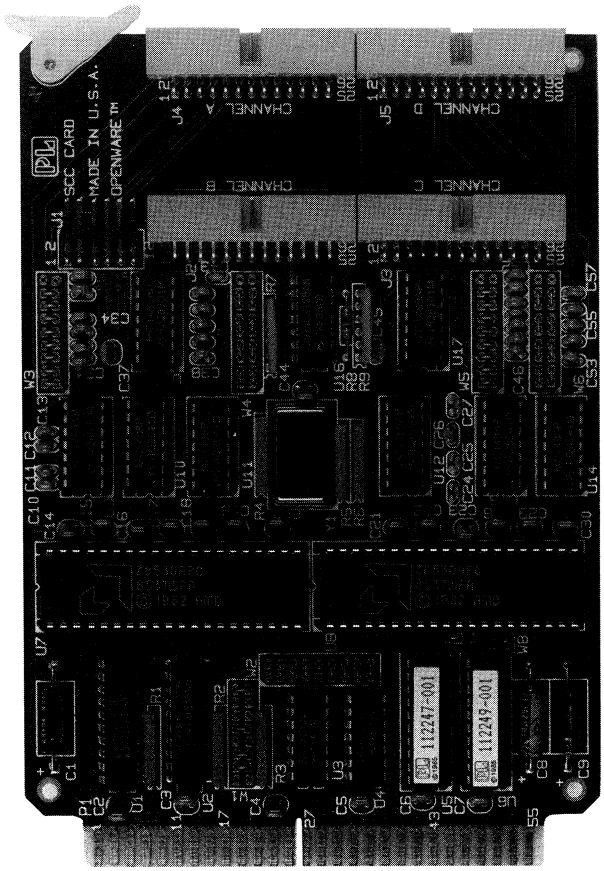


Figure 1-1. 7314A Serial Communications Controller Card

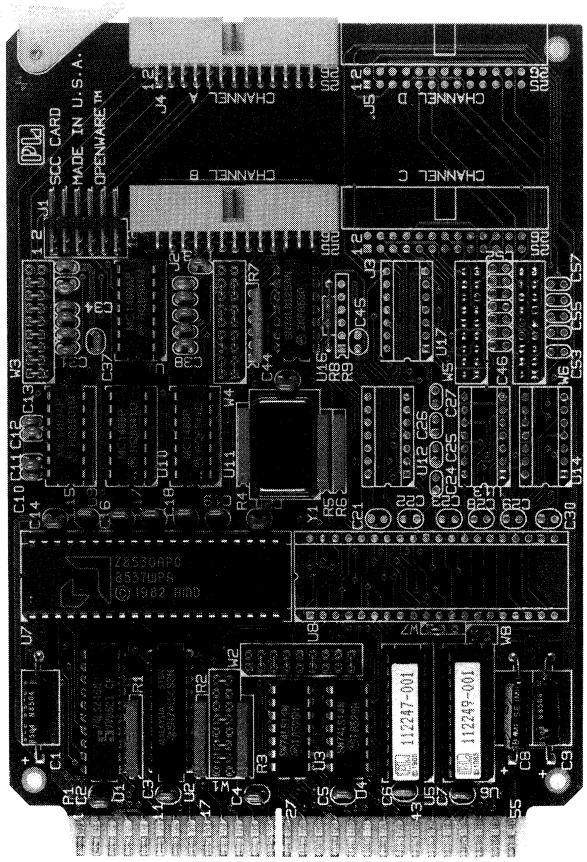


Figure 1-2. 7312 Serial Communications Controller Card

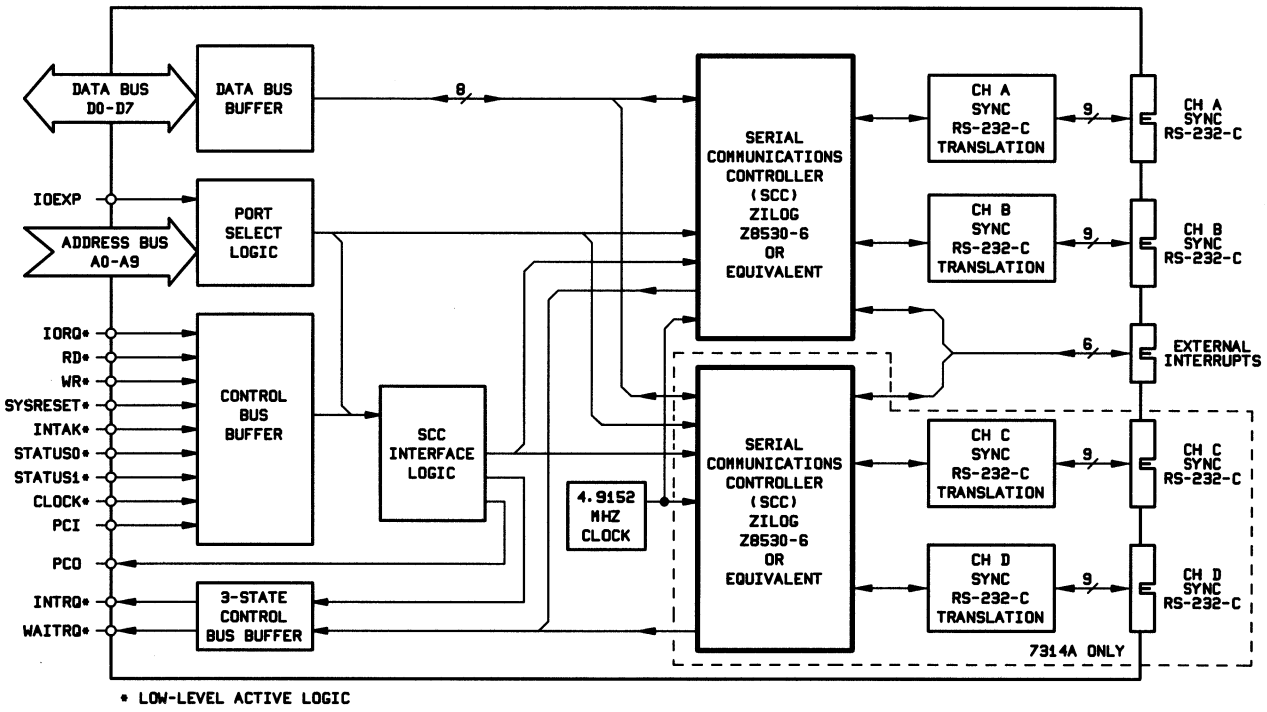


Figure 1-3. Block Diagram

FUNCTIONAL CAPABILITY

Buffering

Data and Control Bus buffers increase noise immunity and reduce the loading on the STD BUS. The INTRQ* and WAITRQ* signals are 3-stated when not in use to further reduce loading.

RS-232-C Interfaces

Four (7314A) or two (7312) independent RS-232-C channels are provided for synchronous and asynchronous serial communication. Each channel is jumper configurable as data terminal equipment (DTE) or data communications equipment (DCE). Figure 1-4 shows the pinout of each RS-232-C interface connector.

PIN NUMBER	SIGNAL FLOW		PIN NUMBER	SIGNAL FLOW	
SIGNAL					SIGNAL
		2	1		
RCLK (DB)	OUT	4	3	IN/OUT	TXD*(BA)
		6	5	IN/OUT	RXD*(BB)
RCLK (DD)	IN/OUT	8	7	IN/OUT	RTS (CA)
		10	9	IN/OUT	CTS (CB)
		12	11	IN/OUT	DSR (CC)
DTR (CD)	IN/OUT	14	13	OUT	SIGNAL GND (AB)
		16	15		
		18	17		
		20	19		
TCLK (DA)	IN/OUT	22	21		
		24	23		
		26	25		

Figure 1-4. RS-232-C Connectors J2, 3, 4, 5

Serial Communications Controller (SCC)

Each onboard dual-channel serial communications controller provides two independent, full-duplex communications channels. Each channel contains independent baud rate registers and is fully programmable to operate in various asynchronous, byte-synchronous (BISYNC), and bit-synchronous (SDLC, HDLC, CCITT-X.25) modes.

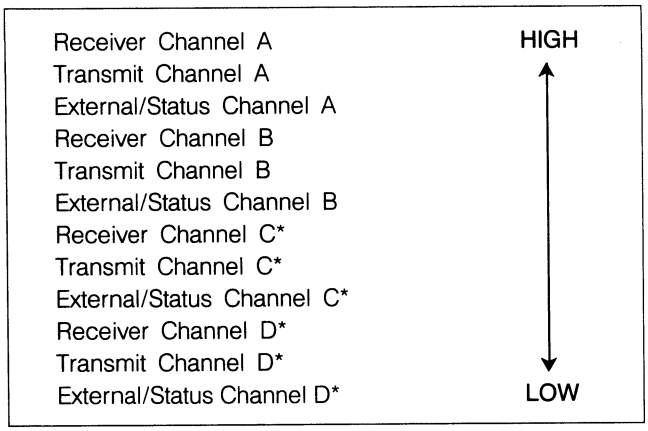
Each serial channel can be programmed to generate standard baud rates from 50 to 19.2K baud. The baud rate may be set higher (up to 78.6K), however RS-232-C specifies the maximum baud rate to be 20K baud. If four channels are operating simultaneously (7314A), the maximum baud rate possible for each channel is software dependent.

The SCC interface logic contained in the two PALs allows the SCC to interface with the 8088, Z80, and 8085 CPUs. When operating (with an 8088 CPU) the 7314A-1 or 7312-1 automatically inserts one WAIT state for every I/O cycle. When operating with a Z80 CPU card, the 7314A-2 and 7312-2 automatically insert one WAIT state during an interrupt acknowledge cycle. No additional WAIT states are required for operation with an 8085 CPU card.

PIN NUMBER	SIGNAL FLOW		PIN NUMBER	SIGNAL FLOW	
SIGNAL					SIGNAL
GND	IN	2	1	IN	INTRQA*
GND	IN	4	3	IN	INTRQB*
GND	IN	6	5	IN	INTRQC*[1]
GND	IN	8	7	IN	INTRQD*[1]
GND	OUT	10	9	OUT	INTRQ1*
GND	OUT	12	11	OUT	INTRQ2*[1]

*Low-level active logic
[1]7314A only

Figure 1-5. Interrupt Request Connector J1



*7314A only

Figure 1-6. SCC Interrupt Source Priority

Interrupt Prioritization

Each of the SCCs can generate an interrupt request. These interrupt requests can either drive the STD BUS INTRQ* line or are available at the user connector. The INTRQ* line is used in the serial or polled interrupt prioritization techniques discussed in the STD BUS Practice section of the Pro-Log *Technical Manual & Data Book,* February 1985 or later edition. The user connector provides a means to connect interrupts to another card for the parallel prioritization method described in the STD BUS Practice. Figure 1-5 shows the pinout for the Interrupt Request Connector.

If using a serial prioritization scheme each channel of the SCCs can generate four unique prioritzed interrupt vectors, dependent upon various SCC conditions as shown in Figure 1-6. Each SCC has an internal priority chain. Additionally, SCC #1 (channels A and B) has higher priority than SCC #2 (channels C and D of the 7314A). The most common way to operate the SCC is to use the Status Affects Vector feature and program the CPU (Z80 or 8088) for vectored interrupts.

Other interrupting cards in the system that do not provide vectors can use the 7314A and 7312 to generate a unique vector. If a channel is programmed to operate asynchronously, it can also be programmed to provide vectors or restart instructions for off-card interrupts by using the interrupt request lines from other cards as inputs to an on-card user connector.

I/O PORT ADDRESS	OUTPUT FUNCTION	INPUT FUNCTION
B0	Channel A (Ch A, SCC#1) XMT Data	Channel A RCV Data
B1	Channel A (Ch A, SCC#1) Control Registers	Channel A Status Registers
B2	Channel B (Ch B, SCC#1) XMT Data	Channel B RCV Data
B3	Channel B (Ch B, SCC#1) Control Registers	Channel B Status Registers
B4*	Channel C (Ch A, SCC#2) XMT Data	Channel C RCV Data
B5*	Channel C (Ch A, SCC#2) Control Registers	Channel C Status Registers
B6*	Channel D (Ch B, SCC#2) XMT Data	Channel D RCV Data
B7*	Channel D (Ch B, SCC#2) Control Registers	Channel D Status Registers

*Not valid for the 7312

Figure 1-7. I/O Port Addressing

In systems that do not require the 7314A or 7312 to generate vectored interrupts, the card can be jumpered to ignore interrupt acknowledge.

I/O Port Mapping

The 7314A occupies eight consecutive, jumper-selectable I/O addresses (two for each channel). The 7312 also occupies eight consecutive addresses; four are used by the card, and four are not valid. As shipped, the 7314A-1 and 7312-1 (8088 compatible versions) decode 10 I/O address lines (A0-A9) which allows them to be mapped anywhere within the 1024-port addressing range available to an 8088 processor.

Although the 8088 CPU can directly access up to 64K I/O ports, the 7314A-1 and 7312-1 decode a 10-bit I/O address as does the IBM PC™.

The 8088 STD BUS Specification defines IOEXP as an enable signal for cards that only decode an 8-bit I/O address. On the 7314A and 7312, IOEXP is jumper-selectable to be active high, active low, or don't care. As shipped, IOEXP is jumpered as don't care.

The 7314A-2 and 7312-2 (Z80 and 8085 compatible versions) are configured to respond to an 8-bit I/O address and can be mapped anywhere within the 256-port addressing range of 8-bit processors.

The STD BUS I/O expansion line (IOEXP) can be used to double the addressing range from 256 to 512 ports when using an 8-bit I/O address in Z80 or 8085 systems.

™IBM PC is a registered trademark of International Business Machines Corporation.

SPECIFICATIONS

Electrical & Environmental

Absolute Maximum Ratings Over Free-Air Temperature Range (Unless Otherwise Noted)

- Supply Voltage, VCC: 0 to 7V
- Auxiliary Supply Voltage, V AUX+: 0 to 15V
- Auxiliary Supply Voltage, V AUX-: 0 TO -15V
- Operating Free-Air Temperature Range: 0°C to +65°C
- Storage Temperature Range: -40°C to +85°C
- Noncondensing Relative Humidity: 5% to 95%

Operating Conditions

		MIN	TYP.	MAX	UNIT
VCC	Supply Voltage	4.75	5.00	5.25	V
V AUX+		11.40	12.00	12.60	V
V AUX-		-11.40	-12.00	-12.60	V
Operating Free-Air Temperature		0	25	65	°C
Noncondensing Relative Humidity		5	—	95	%

Electrical Characteristics Over Operating Free-Air Temperature Range (Unless Otherwise Noted)

		7314A			7312			
Parameter	Test Conditions	MIN	TYP	MAX	MIN	TYP	MAX	UNIT
*ICC	(VCC)	—	910	1260	—	670	930	mA
*I AUX+	(V AUX+)	—	75	120	—	38	60	mA
*I AUX-	(V AUX-)	—	-95	-145	—	48	73	mA

*Unused RS-232-C drivers and receivers may be removed to reduce power consumption.

All logic signal levels related to the 7314A and 7312 that appear on the bus conform to the STD BUS TTL requirements as specified in the general *STD Series 7000 Specification,* document #111684.

7314A-1 and -2 MTBF is 26 years. The 7312-1 and -2 MTBF is 29 years. MTBF is calculated using component standard failure rates specified in *Failure Rates & MTBF Calculation,* document #111443.

Both the Series *7000 and 70C00 Specifications* and *Failure Rates & MTBF Calculations* are included in Sections 3 and 4, respectively, of the *Technical Manual and Data Book, February 1985 or later,* Pro-Log document #107855.

RS-232-C Serial Switching Characteristics Over Operating Range

Line Driver (MC1488)

0°C ≤ T ≤ 65°C
V AUX+ + 12V, V AUX- = -12V
All typical values are at T = 25°C

Parameter	Test Conditions	MIN	TYP.	MAX	UNIT
VOH	VIL=.8V; R1=3K	8.2	9.5	—	V
VOL	VIH=1.9V; R1=3K	-8.1	-9.4	—	V
IOS(H)	VI=.8V; VO=0	-6.0	-10.0	-12.0	mA
IOS(L)	VI=1.9V; VO=0	6.0	10.0	12.0	mA
RO	V AUX+=0; V AUX-=0; -2V≤ VO ≤ 2V	300			0hm

Line Receiver (MC1489A)

0°C ≤ T ≤ 65°C
VCC = 5V
All typical values are at T = 25°C and VCC = 5V

Parameter	Test Conditions	MIN	TYP.	MAX	UNIT
VIH	TA=+25°C, VOL< 0.45V	3.0	—	—	V
VIL	TA=+25°C, VOH> 2.5V, IL=-0.5mA	—	—	-3.0	V
RIN	TA=+25°C	—	7K	—	ohm

*RS-232-C Maximum Baud Rates**

- Asynchronous: 19.2K
- Synchronous: 19.2K

*The asynchronous and synchronous baud rates may be set up to 78.6K, but the RS-232-C specifies a maximum baud rate of 20K.

Timing Specifications

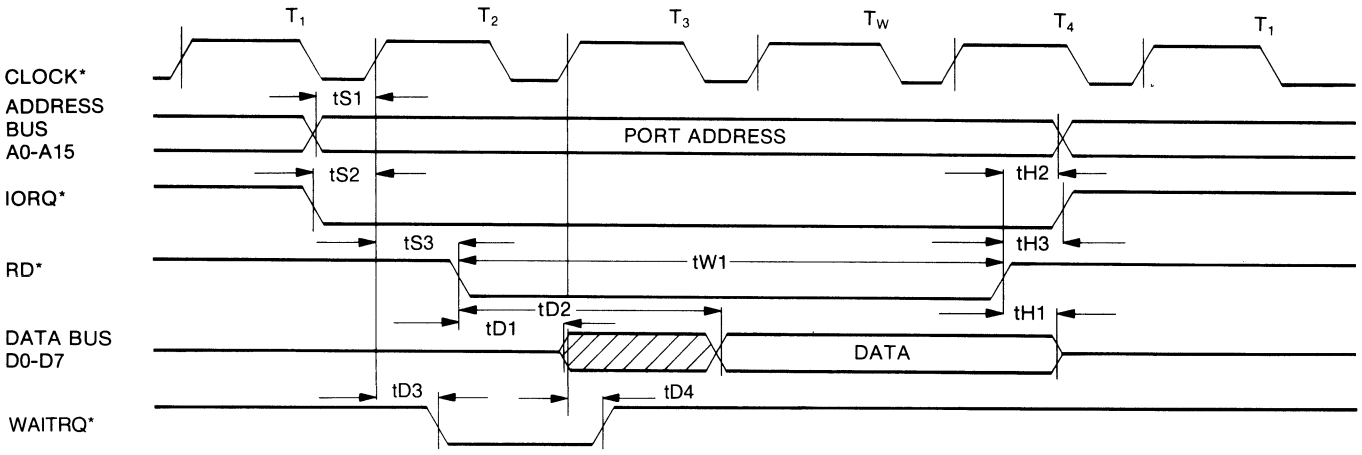
The critical timing parameters as defined by the proposed IEEE P961/D4 STD BUS standard (February 4, 1985 document, Figures 9 & 10) are listed below. The 7314A and 7312 meet the STDMG STD-8088, the September 10, 1984 STDMG STD-Z80, and May 15, 1984 STDMG STD-8085/80C85A specifications.

	8088 7314A-1, 7312-1	Z80 7314A-2, 7312-2	8085 7314A-2, 7312-2
tAR	225	245	245
tSWD	245	250	120
tHWD	40	40	40

Critical Timing Parameters (in nanoseconds)

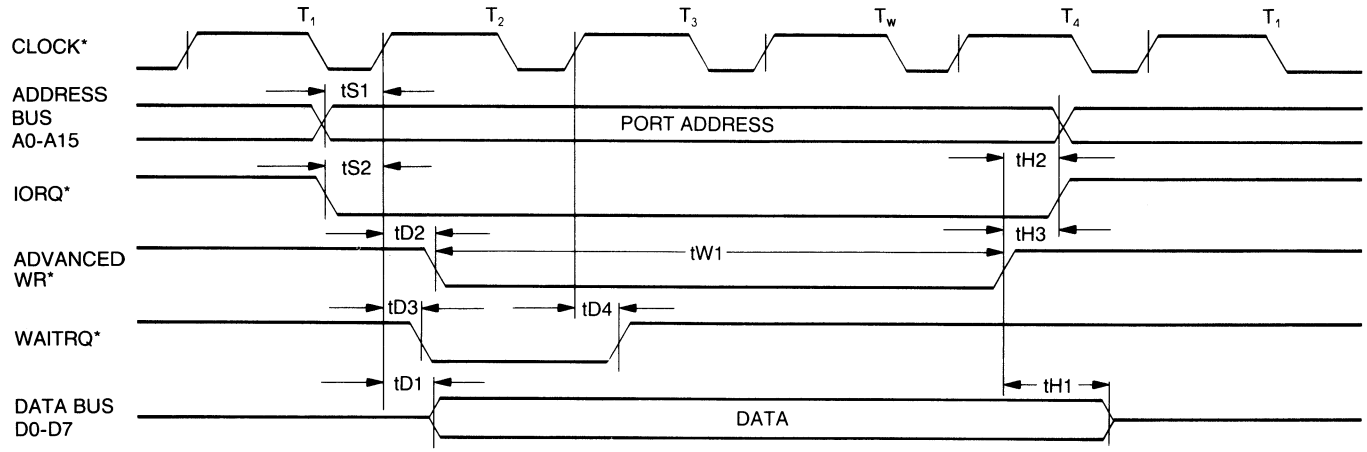
The following diagrams present the timing specifications of the 7314A and 7312 cards operating under the following parameters:

0°C ≤ T ≤ +65°C
4.75V ≤ VCC ≤ 5.25V



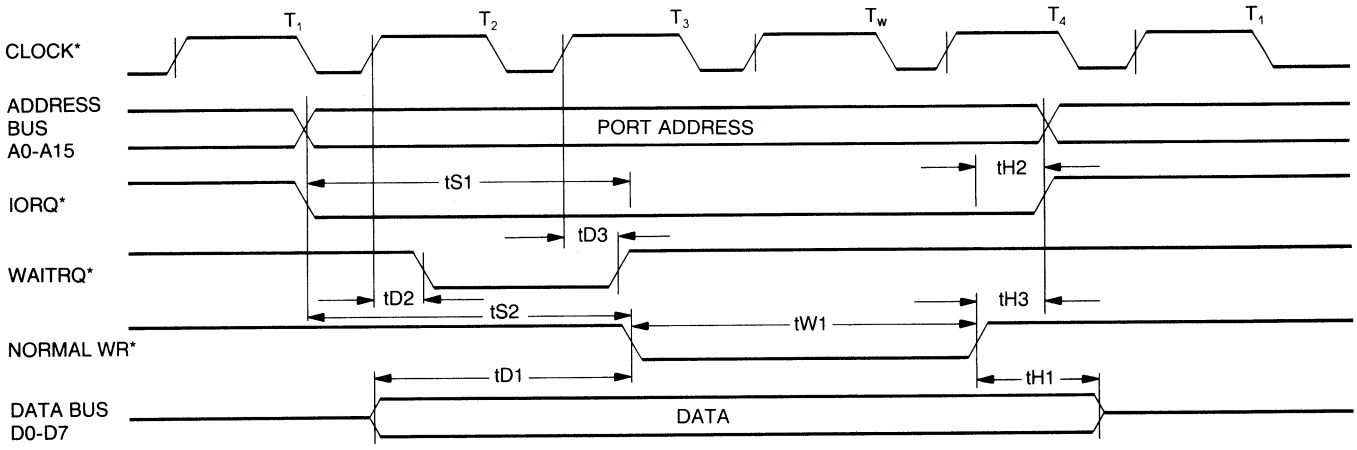
PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	RD* low to data bus driven		105
tD2	RD* low to data bus valid		225
tD3	T2 CLOCK* high to WAITRQ* low		80
tD4	T3 CLOCK* high to WAITRQ* high		75
tH1	RD* high to data bus float		85
tH2	Address hold after RD* high	0	
tH3	IORQ* low after RD* high	0	
tS1	Address valid to T2 CLOCK* high	0	
tS2	IORQ* low to T2 CLOCK* high	0	
tS3	T2 CLOCK* high to RD* low	0	
tW1	RD* low pulse width	375	

Figure 1-8. 7314A-1, 7312-1 Input Timing with an 8088 CPU



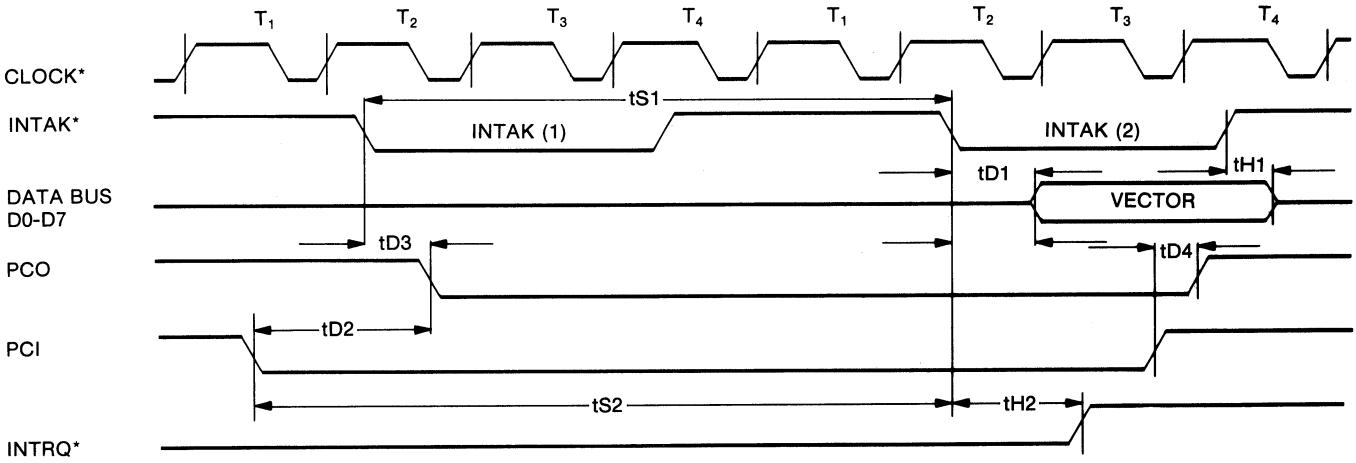
PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	Data valid after T_2 CLOCK* high		125
tD2	WR* low after T_2 CLOCK* high	0	
tD3	WAITRQ* low after T_2 CLOCK* high		80
tD4	WAITRQ* high after T_3 CLOCK* high		75
tH1	Data hold after WR* high	40	
tH2	Address hold after WR* high	35	
tH3	IORQ* low after WR* high	20	
tS1	Address valid before T_2 CLOCK* high	0	
tS2	IORQ* low before T_2 CLOCK* high	0	
tW1	WR* pulse width	370	

Figure 1-9. 7314A-1, 7312-1 Output Timing with an 8088 CPU, Advanced Write



PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	Data valid before WR* low	40	
tD2	WAITRQ* low after T_2 CLOCK* high		80
tD3	WAITRQ* high after T_2 CLOCK* high		75
tH1	Data hold after WR* high	40	
tH2	Address hold after WR* high	35	
tH3	IORQ* low after WR* high	20	
tS1	Address valid before WR* low	95	
tS2	IORQ* low before WR* low	0	
tW1	Write pulse width	250	

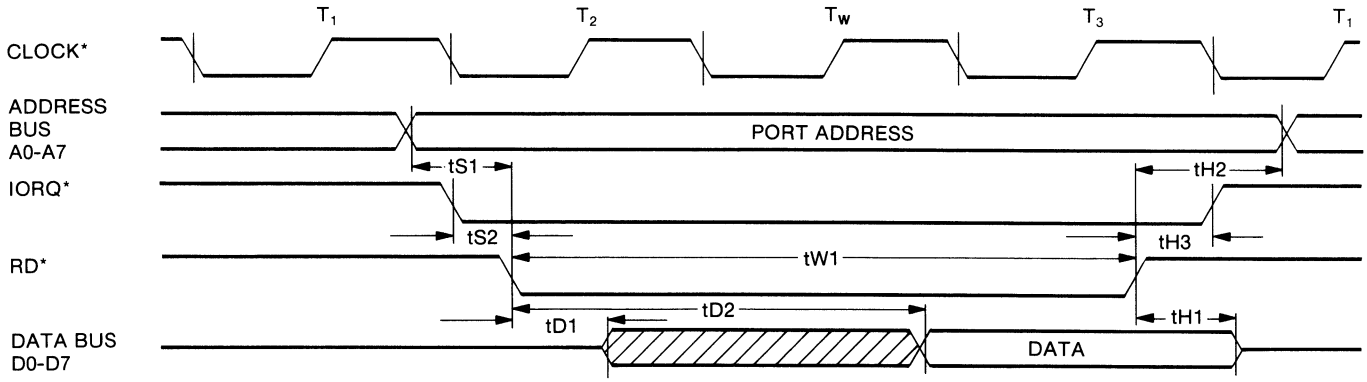
Figure 1-10. 7314A-1, 7312-1 Output Timing with an 8088 CPU, Normal Write



PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	INTAK*(2) low to vector valid		225
tD2	PCI low to PCO low		52
tD3	INTAK*(1) low to PCO low (interrupt occurs just before INTAK*(1) low)		275
tD4	PCI high to PCO high		52
tH1	INTAK*(2) high to data bus float		85
tH2	INTAK*(2) low to INTRQ* high		570
tS1	INTAK*(1) low to INTAK*(2) low	(1)	(1)
tS2	PCI low to INTAK*(2) low (7314A/7312 is lowest priority)	155	

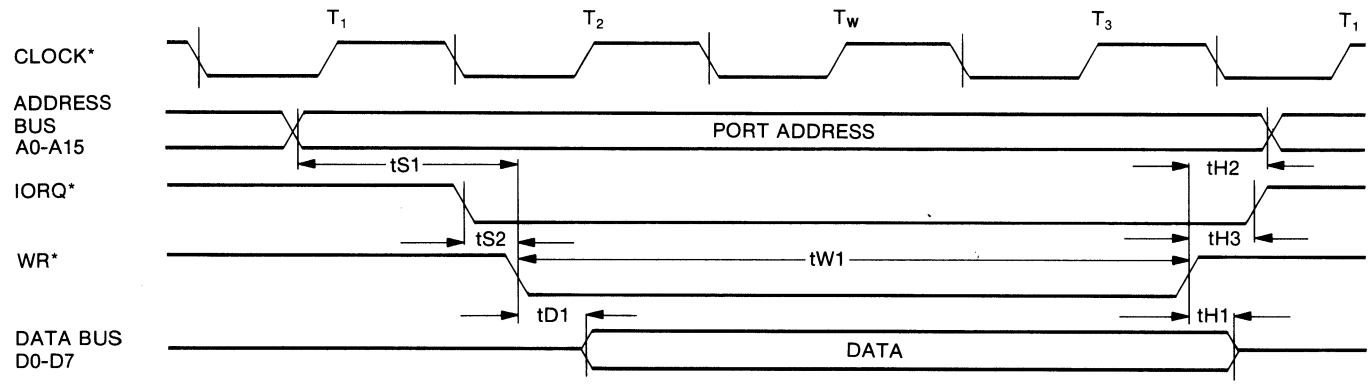
(1) This represents the priority chain settling time, and is system-dependent.

Figure 1-11. 7314A-1, 7312-1 Interrupt Acknowledge Timing with an 8088 CPU



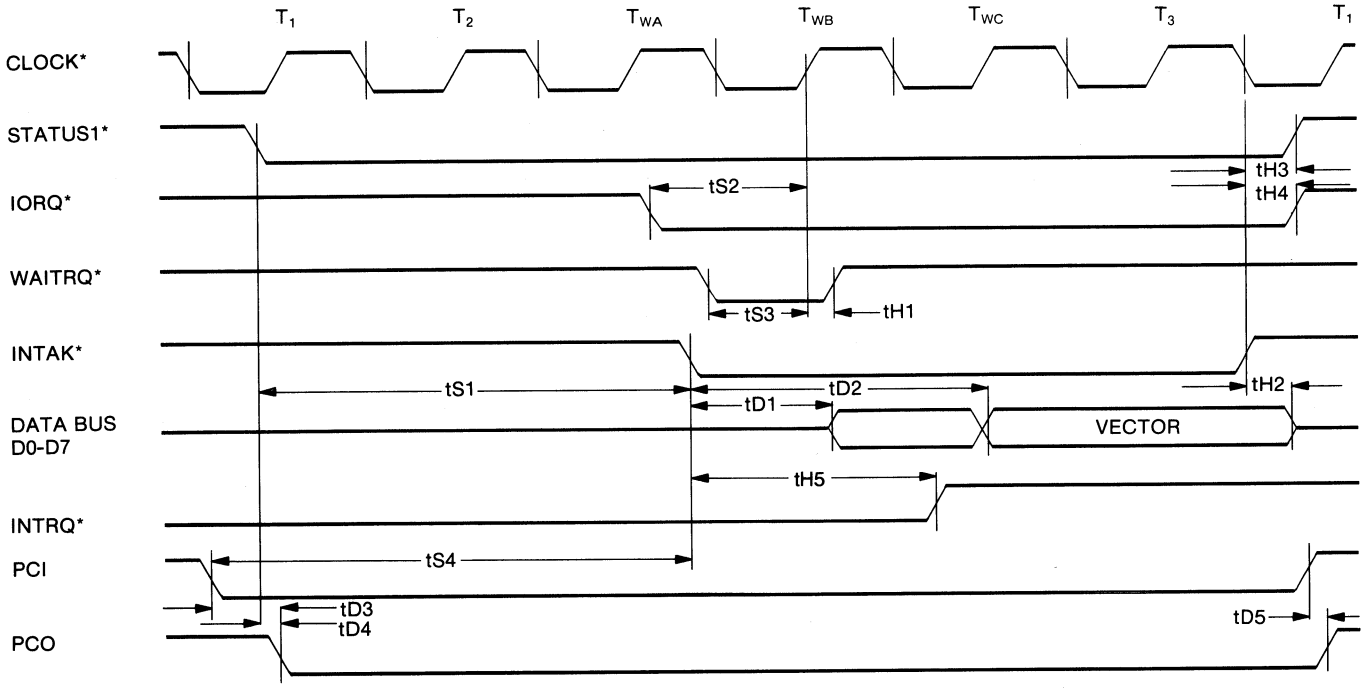
PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	RD* low to data bus driven		130
tD2	RD* low to data bus valid		245
tH1	RD* high to data bus float		85
tH2	Address hold after RD* high	0	
tH3	IORQ* low after RD* high	0	
tS1	Address valid to RD* low	85	
tS2	IORQ* low to RD* low	0	
tW1	RD* low pulse width	250	

Figure 1-12. 7314A-2, 7312-2 Input Timing With a Z80 CPU



PARAMETER	DESCRIPTION	NANOSECONDS			
		4.0 MHz		2.5 MHz	
		MIN	MAX	MIN	MAX
tD1	WR* low before data valid		40		100
tH1	Data hold after WR* high	40		40	
tH2	Address hold after WR* high	35		35	
tH3	IORQ* low after WR* high	20		20	
tS1	Address valid to WR* low	95		95	
tS2	IORQ* low to WR* low	0		0	
tW1	WR* low pulse width	290		350	

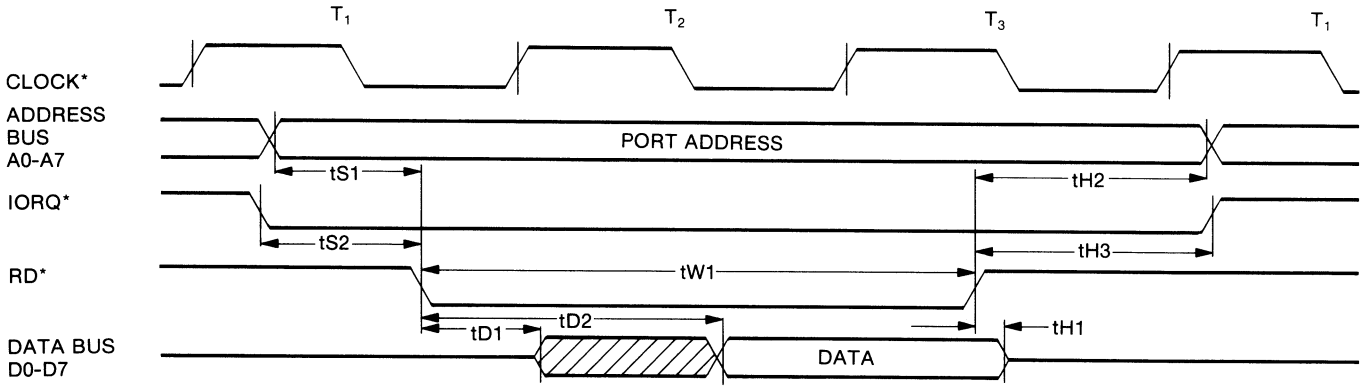
Figure 1-13. 7314A-2, 7312-2 Output Timing With a Z80 CPU



PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	INTAK* low to data bus driven		105
tD2	INTAK* low to data bus valid		250
tD3	PCI low to PCO low		25
tD4	STATUS1* low to PCO low (interrupt occurs just before STATUS1*)		275
tD5	PCI high to PCO high		52
tH1	WAITRQ* hold after TWB CLOCK* high		55
tH2	INTAK* high to data bus float		85
tH3	STATUS1* hold after INTAK* high	0	
tH4	IORQ* hold after INTAK* high	0	
tH5	INTAK* low to INTRQ* high		570
tS1	STATUS1* low to INTAK* low	(1)	(1)
tS2	IORQ* low to TWB CLOCK* high	165	
tS3	WAITRQ* low to TWB CLOCK* high	110	
tS4	PCI low to INTAK* low (7314A/7312 is lowest priority)	155	

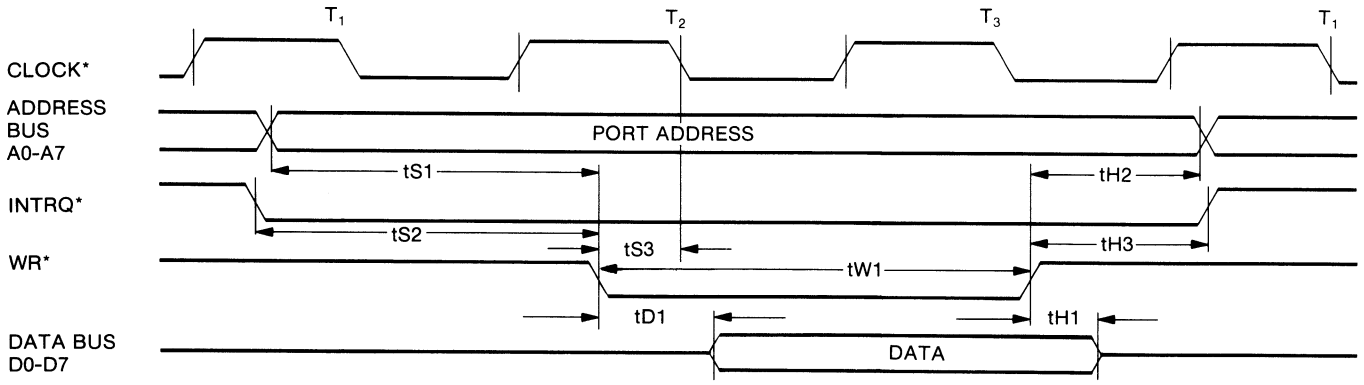
(1) This represents the priority chain settling time, and is system dependent.

Figure 1-14. 7314A-2, 7312-2 Interrupt Acknowledge Timing with a Z80 CPU



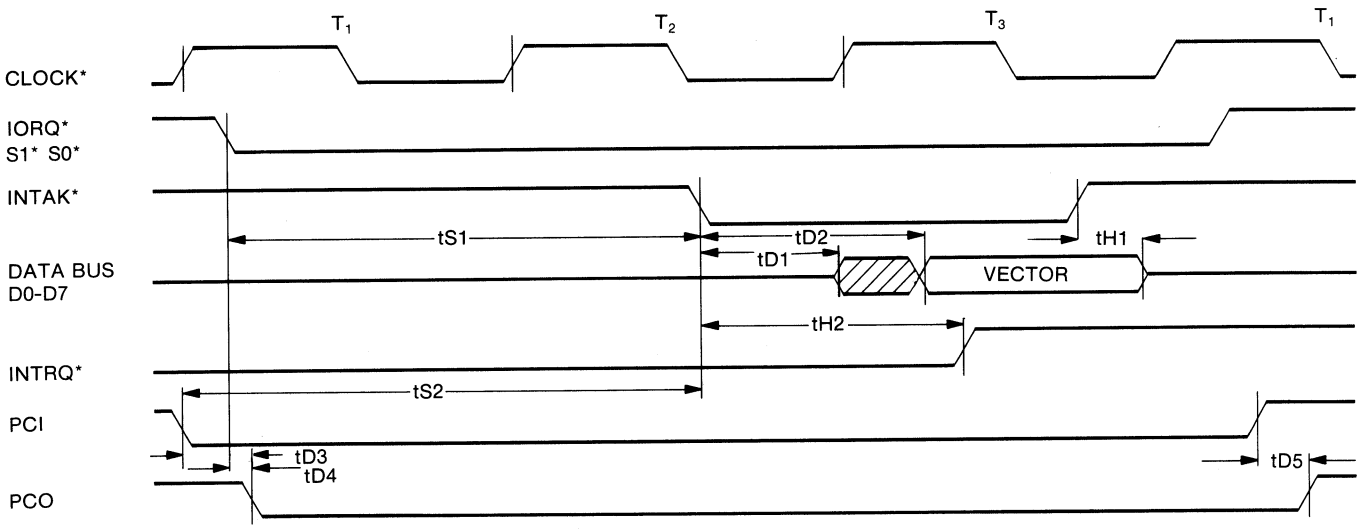
PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	RD* low to data bus driven		130
tD2	RD* low to data bus valid		245
tH1	RD* high to data bus float		85
tH2	Address hold after RD* high	0	
tH3	IORQ* low after RD* high	0	
tS1	Address valid to RD* low	85	
tS2	IORQ* low to RD* low	0	
tW1	RD* low pulse width	250	

Figure 1-15. 7314A-2, 7312-2 Input Timing With an 8085 CPU



PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	Data valid before WR* low	130	
tH1	Data hold after WR* high	40	
tH2	Address hold after WR* high	35	
tH3	IORQ* low after WR* high	20	
tS1	Address valid before WR* low	95	
tS2	IORQ* low to WR* low	0	
tS3	WR* low to T2 low	0	
tW1	WR* low pulse width	250	

Figure 1-16. 7314A-2, 7312-2 Output Timing With an 8085 CPU



PARAMETER	DESCRIPTION	NANOSECONDS	
		MIN	MAX
tD1	INTAK* low to data bus driven		130
tD2	INTAK* low to data bus valid		245
tD3	PCI low to PCO low		52
tD4	IORQ*, S1*, S0* low to PCO low (interrupt occurs just before IORQ*, S1*, S0* low)		52
tD5	PCI high to PCO high		25
tH1	INTAK* high to data bus float		85
tH2	INTAK* low to INTRQ* high		570
tS1	IORQ*, S1*, S0* low to INTAK* low	(1)	(1)
tS2	PCI low to INTAK* low (7314A/7312 is lowest priority)	155	

(1) This represents the priority chain settling time, and is system-dependent.

Figure 1-17. 7314A-2, 7312-2 Interrupt Acknowledge Timing with an 8085A CPU

Mechanical Specifications

The 7314A and 7312 meet all general mechanical specifications of the STD BUS Series 7000 as given in the *STD Series 7000 and 70C00 Specification,* document #111684. This document is included in the *Technical Manual and Data Book,* Feb. '85 or later, Pro-Log document #107855.

Card Spacing

Requires one .5-inch (12.7 mm) card slot.

PIN NUMBER	OUTPUT (LSTTL DRIVE)	INPUT (LSTTL LOADS)		PIN NUMBER	OUTPUT (LSTTL DRIVE)	INPUT (LSTTL LOADS)	
MNEMONIC							MNEMONIC
+5V	IN		2	1		IN	+5V
GROUND	IN		4	3		IN	GROUND
-5V			6	5			-5V
D7	1	55	8	7	55	1	D3
D6	1	55	10	9	55	1	D2
D5	1	55	12	11	55	1	D1
D4	1	55	14	13	55	1	D0
A15			16	15		1	A7
A14			18	17		1	A6
A13			20	19		1	A5
A12			22	21		1	A4
A11			24	23		1	A3
A10			26	25		1	A2
A9	1		28	27		1	A1
A8	1		30	29		1	A0
RD*	1		32	31		1	WR*
MEMRQ*			34	33		1	IORQ*
MEMEX			36	35		1	IOEXP
MCSYNC*	1[1]		38	37			REFRESH*
STATUS 0*	1[2]		40	39		1[2]	STATUS 1*
BUSRQ*			42	41			BUSAK*
INTRQ*		25	44	43		1	INTAK*
NMIRQ*			46	45	55		WAITRQ*
PBRESET*			48	47		1	SYSRESET*
CNTRL*			50	49		1	CLOCK*
PCI	1		52	51	55		PCO
AUX GND			54	53			AUX GND
AUX -V	IN		56	55		IN	AUX +V

* Low-level active logic
One LSTTL load = -0.4 mA
[1] 7314A-1, 7312-1
[2] 7314A-2, 7312-2

Figure 1-18. STD Edge Connector

Cables for Use With the 7314A and 7312

7314A, 7312 Connector	Pro-Log Cable	Description
J2, J3, J4, J5 RS-232-C	RC701-3 or or RC702-3	3 feet (914 mm), terminates in 25-pin female (RC701-3) or male (RC702-3) D connector with RS-232-C pinout.
J1 Interrupt Request	RC704-1 or RC704-2	12-inch (304.8 mm) (RC704-1) or 24-inch (304.8 mm) (RC704-2) twisted-pair, general-purpose interconnect cable.

MATING CONNECTORS FOR THE 7314A, 7312 CONNECTOR		
Description	Robinson Nugent Part #	T&B/Ansley Part #
26-Pin Transition	IDS-26PK-SR-G30	609-2630-1

Figure 1-19. Cables and Connectors

REFERENCE SPECIFICATIONS

EIA Standard RS-232-C
Zilog, AMD, and Intel Z8530 SCC Product Specification

CARD LAYOUT AND JUMPERS

Several optional functions on the 7314A and 7312 are selected by wire jumpers.

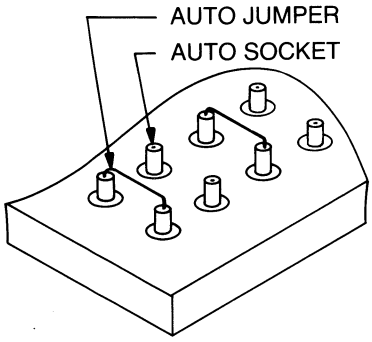
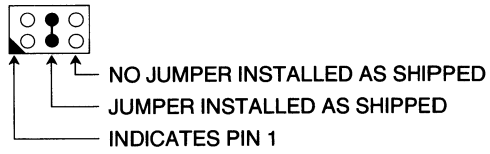


Figure 2-2. Jumper Configuration

The shorting wires can be removed and installed without the use of solder equipment, as shown in Figure 2-2. Figure 2-3 lists sources for the shorting wires.

SOURCE	PART NUMBER
Fancourt Industries, Inc.	J 0.100X0.250 B 24
Squires Electronics, Inc.	J 0.100X0.250 B 24

Figure 2-3 Sources for Shorting Jumpers



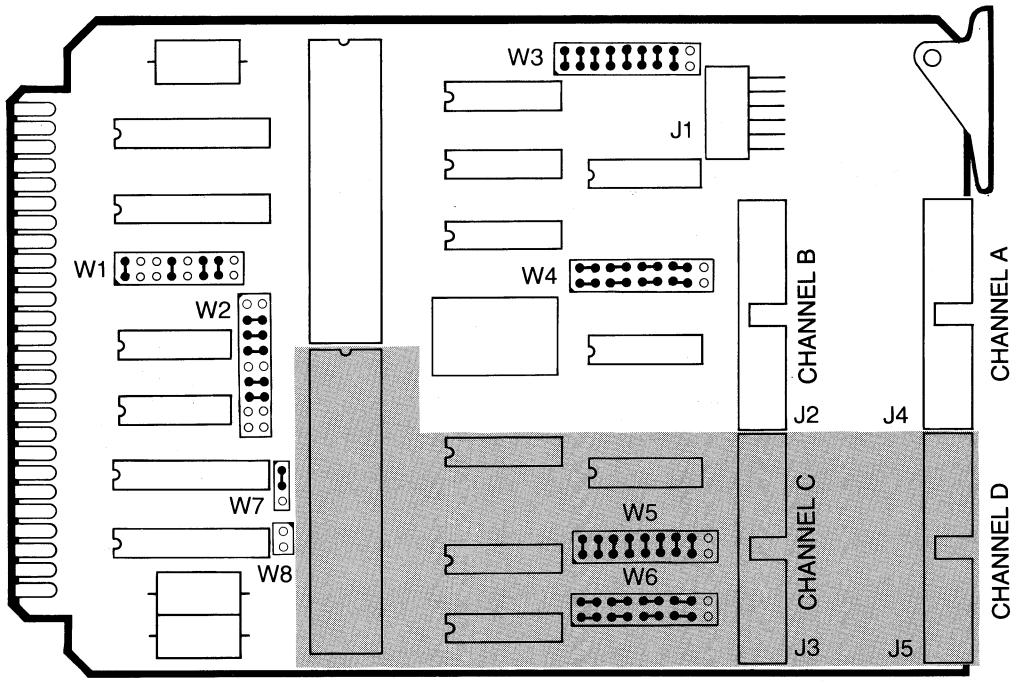


Figure 2-4. Physical Layout and Jumper Options

JUMPER GROUP	DESCRIPTION		AS SHIPPED			
			7314A-1	7314A-2	7312-1	7312-2
W1 Page 2-4	1-2 No jumper 3-16	IOEXP active low IOEXP active high I/O Port Mapping (B0-B7 as shipped)	No jumper 3-4,5-6 9-10,15-16	No jumper 9-10,15-16	No jumper 3-4,5-6 9-10,15-16	No jumper 9-10,15-16
W2 Page 2-4	1-2 No jumper	IOEXP enabled IOEXP don't care	No jumper	No jumper	No jumper	No jumper
	3-4, 5-6 No jumper	A9, A8 enabled for 10-bit I/O port addressing A9, A8 disabled for 8-bit I/O port addressing	3-4, 5-6	No jumper	3-4, 5-6	No jumper
Page 2-8	7-8 9-10	Interrupt vector response enabled Interrupt vector response disabled	7-8	7-8	7-8	7-8
	11-12 No jumper	INTRQ* connected to STD BUS INTRQ* disconnected from STD BUS	11-12	11-12	11-12	11-12
	13-14 No jumper	INTRQ* from both SCCs wire-ORed: STD INTRQ*=J1-9=INTRQ1*+INTRQ2* STD INTRQ*=J1-9=INTRQ2*	13-14	13-14	No jumper (Not implemented on the 7312 card)	No jumper (Not implemented on the 7312 card)
	15-16 No jumper	7314A-1, -2 7312-1, -2	15-16	15-16	No jumper	No jumper
	17-18 No jumper	7314A-1, 7312-1: Advanced write 7314A-2, 7312-2: Z80 CPU 7314A-1, 7312-1: Normal write 7314A-2, 7312-2: 8085 CPU	17-18	17-18	17-18	17-18
W3 Page 2-8	1-16 17-18	Selects RS-232-C interface for Channel A (J4) Provides a transmit clock as a DTE input	1-2, 3-4, 5-6, 7-8,9-10, 11-12, 13-14, 15-16 (DCE) No jumper	1-2, 3-4, 5-6, 7-8,9-10, 11-12, 13-14, 15-16 (DCE) No jumper	1-2, 3-4, 5-6, 7-8,9-10, 11-12, 13-14, 15-16 (DCE) No jumper	1-2, 3-4, 5-6, 7-8,9-10, 11-12, 13-14, 15-16 (DCE) No jumper
W4 Page 2-8	1-16 17-18	Selects RS-232-C interface for channel B (J2) Provides a transmit clock as a DTE input.	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper
W5 Page 2-8	1-16 17-18	Selects RS-232-C interface for channel C (J3) Provides a transmit clock as a DTE input.	1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14, 15-16 (DCE) No jumper	1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14, 15-16 (DCE) No jumper	Channels C and D	Channels C and D
W6 Page 2-8	1-16 17-18	Selects RS-232-C interface for channel D (J5) Provides a transmit clock as a DTE input.	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper	1-3, 2-4, 5-7, 6-8, 9-11, 10-12, 13-15, 14-16 (DTE) No jumper	are not implemented on the 7312 card	are not implemented on the 7312 card
W7	1-2 2-3	7314A-1, 7312-1 7314A-2, 7312-2	1-2	2-3	1-2	2-3
W8	1-2 No jumper	7314A-2, 7312-2 7314A-2, 7312-2	No jumper	1-2	No jumper	1-2

Figure 2-4. (Continued)

I/O ADDRESS MAPPING

In normal operation, the 7314A and 7312 are addressed directly by the processor card. The input and output ports respond to a jumper-selected combination of address lines, I/O Request (IORQ*), I/O Expansion (IOEXP), and either Read (RD*) or Write (WR*). As shipped, the 7314A-1 and 7312-1 decode address lines A0 through A9, and the 7314A-2 and 7312-2 decode address lines A0 through A7.

The 7314A and 7312 each occupy eight consecutive I/O addresses regardless of their mapping assignments. The card is shipped with the eight ports mapped at hex address B0 through B7. You can retain these addresses or change them by moving the installed jumpers in the W1 header. While each card's port addresses are generally arbitrary, they must differ from all other I/O port addresses in the system. If they do not differ, multiple cards will respond to the same I/O read instruction, resulting in bus contention.

8-Bit Port Addressing Systems

The 7314A-2 and 7312-2 can be remapped anywhere within the 256-port addressing range of 8085 and Z80 processors. To change the I/O addressing, move the jumpers in the W1 header. W1 3-4 and 5-6, and W2 3-4 and 5-6 are not installed in the 7314A-2 and 7312-2. Figure 2-5 shows jumper positions for various 8-bit port addresses.

The I/O expansion line (IOEXP) of the STD BUS allows a system to access 512 I/O ports rather than the standard 256. Pro-Log processor cards typically connect the IOEXP line to ground. The 7314A-2 and 7312-2 can be enabled with IOEXP low by installing jumper W1 1-2 and it can be enabled with IOEXP high by removing jumper W1 1-2. To ignore IOEXP, remove the jumpers at W1 1-2 and W2 1-2. As shipped, the 7314A-2 and 7312-2 are enabled with IOEXP don't care (W1 1-2 and W2 1-2 removed).

ADDRESS RANGE	CARD SELECT DECODE HEADER W1[1]					
	1-2[2]	7-8	9-10	11-12	13-14	15-16
00-07		X	X	X	X	X
08-0F		X	X	X	X	
10-17		X	X	X		X
18-1F		X	X	X		
20-27		X	X		X	X
28-2F		X	X		X	
30-37		X	X			X
38-3F		X	X			
40-47		X		X	X	X
48-4F		X		X	X	
50-57		X		X		X
58-5F		X		X		
60-67		X			X	X
68-6F		X			X	
70-77		X				X
78-7F		X				
80-87			X	X	X	X
88-8F			X	X	X	
90-97			X	X		X
98-9F			X	X		
A0-A7			X		X	X
A8-AF			X		X	
B0-B7			X			X [3]
B8-BF			X			
C0-C7				X	X	X
C8-CF				X	X	
D0-D7				X		X
D8-DF				X		
E0-E7					X	X
E8-EF					X	
F0-F7						X
F8-FF						

X=Jumper installed
[1]W1 3-4 and 5-6 are not installed.
[2]IOEXP jumper—used for alternate 256 I/O port mapping
[3]As shipped

Figure 2-5. 8-Bit Port Address Mapping

Ten-Bit Port Addressing

As in the IBM PC, a 10-bit I/O port address is decoded during each I/O operation. The 7314A-1 and 7312-1 decode address lines A0-A9, which allows them to be mapped anywhere within a 1024 port addressing range. Figure 2-6 shows how the 7314A-1 and 7312-1 are mapped in a typical 10-bit port addressing system, with IOEXP not used.

ADDRESSES A9, A8 / ADDRESSES A7-A0	00	01	10	11
00 | 0F				
10 | 1F				
20 | 2F				
30 | 3F				
40 | 4F				
50 | 5F				
60 | 6F				
70 | 7F				
80 | 8F				
90 | 9F				
A0 | AF				
B0 | BF	7314A-1, 7312-1			
C0 | CF				
D0 | DF				
E0 | EF				
F0 |				

Figure 2-6. 1K I/O Port Address Map

Figure 2-7 shows jumper positions for various 10-bit port addresses.

ADDRESS RANGE	CARD SELECT DECODE HEADER W1							
	1-2[1]	3-4	5-6	7-8	9-10	11-12	13-14	15-16
0000-0007		X	X	X	X	X	X	X
0008-000F		X	X	X	X	X	X	
0010-0017		X	X	X	X	X		X
0018-001F		X	X	X	X	X		
0020-0027		X	X	X	X		X	X
0028-002F		X	X	X	X		X	
0030-0037		X	X	X	X			X
0038-003F		X	X	X	X			
0040-0047		X	X	X		X	X	X
0048-004F		X	X	X		X	X	
0050-0057		X	X	X		X		X
0058-005F		X	X	X		X		
0060-0067		X	X	X			X	X
0068-006F		X	X	X			X	
[2] 0070-0077		X	X	X				X
0078-007F		X	X	X				
0080-0087		X	X		X	X	X	X
0088-008F		X	X		X	X	X	
0090-0097		X	X		X	X		X
0098-009F		X	X		X	X		
00A0-00A7		X	X		X		X	X
00A8-00AF		X	X		X		X	
00B0-00B7		X	X		X			X [4]
00B8-00BF		X	X		X			
00C0-00C7		X	X			X	X	X
00C8-00CF		X	X			X	X	
00D0-00D7		X	X			X		X
00D8-00DF		X	X			X		
00E0-00E7		X	X				X	X
00E8-00EF		X	X				X	
00F0-00F7		X	X					X
00F8-00FF		X	X					
0100-0107		X		X	X	X	X	X
0108-010F		X		X	X	X	X	
0110-0117		X		X	X	X		X
0118-011F		X		X	X	X		
0120-0127		X		X	X		X	X
0128-012F		X		X	X		X	
0130-0137		X		X	X			X
0138-013F		X		X	X			
0140-0147		X		X		X	X	X
0148-014F		X		X		X	X	
0150-0157		X		X		X		X
0158-015F		X		X		X		
0160-0167		X		X			X	X
0168-016F		X		X			X	
0170-0177		X		X				X
[3] 0178-017F		X		X				
0180-0187		X			X	X	X	X
0188-018F		X			X	X	X	
0190-0197		X			X	X		X
0198-019F		X			X	X		
01A0-01A7		X			X		X	X
01A8-01AF		X			X		X	
01B0-01B7		X			X			X
01B8-01BF		X			X			
01C0-01C7		X				X	X	X
01C8-01CF		X				X	X	
01D0-01D7		X				X		X
01D8-01DF		X				X		
01E0-01E7		X					X	X
01E8-01EF		X					X	
01F0-01F7		X						X
01F8-01FF		X						

X=Jumper installed

[1]W1 1-2 is the IOEXP jumper used to enable 8-bit I/O port addressing when active low.

[2]These port addresses are redundantly mapped throughout hex 1XXX-FXXX (W1 1-2 removed) and at hex X4XX, X8XX, or XCXX

[3]These port addresses are redundantly mapped throughout hex 1XXX-FXXX (W1 1-2 removed) and at hex X5XX, X9XX, or XDXX

[4]As shipped

Figure 2-7. Ten Bit Port Address Mapping

ADDRESS RANGE	CARD SELECT DECODE HEADER W1							
	1-2[1]	3-4	5-6	7-8	9-10	11-12	13-14	15-16
0200-0207			X	X	X	X	X	X
0208-020F			X	X	X	X	X	
0210-0217			X	X	X	X		X
0218-021F			X	X	X	X		
0220-0227			X	X	X		X	X
0228-022F			X	X	X		X	
0230-0237			X	X	X			X
0238-023F			X	X	X			
0240-0247			X	X		X	X	X
0248-024F			X	X		X	X	
0250-0257			X	X		X		X
0258-025F			X	X		X		
0260-0267			X	X			X	X
0268-026F			X	X			X	
[2] 0270-0277			X	X				X
0278-027F			X	X				
0280-0287			X		X	X	X	X
0288-028F			X		X	X	X	
0290-0297			X		X	X		X
0298-029F			X		X	X		
02A0-02A7			X		X		X	X
02A8-02AF			X		X		X	
02B0-02B7			X		X			X
02B8-02BF			X		X			
02C0-02C7			X			X	X	X
02C8-02CF			X			X	X	
02D0-02D7			X			X		X
02D8-02DF			X			X		
02E0-02E7			X				X	X
02E8-02EF			X				X	
02F0-02F7			X					X
02F8-02FF			X					
0303-0307				X	X	X	X	X
0308-030F				X	X	X	X	
0310-0317				X	X	X		X
0318-031F				X	X	X		
0320-0327				X	X		X	X
0328-032F				X	X		X	
0330-0337				X	X			X
0338-033F				X	X			
0340-0347				X		X	X	X
0348-034F				X		X	X	
0350-0357				X		X		X
0358-035F				X		X		
0360-0367				X			X	X
0368-036F				X			X	
[3] 0370-0377				X				X
0378-037F				X				
0380-0387					X	X	X	X
0388-038F					X	X	X	
0390-0397					X	X		X
0398-039F					X	X		
03A0-03A7					X		X	X
03A8-03AF					X		X	
03B0-03B7					X			X
03B8-03BF					X			
03C0-03C7						X	X	X
03C8-03CF						X	X	
03D0-03D7						X		X
03D8-03DF						X		
03E0-03E7							X	X
03E8-03EF							X	
03F0-03F7								X
03F8-03FF								

X=Jumper installed
[1]W1 1-2 is the IOEXP jumper used to enable 8-bit I/O port addressing when active low
[2]These port addresses are redundantly mapped throughout hex 1XXX-FXXX (W1 1-2 removed) and at hex X6XX, XAXX, or XEXX
[3]These port addresses are redundantly mapped throughout hex 1XXX-FXXX (W1 1-2 removed) and at hex X7XX, XBXX, or XFXX

Figure 2-7. (Continued)

Interrupt Vector Enable

The 7314A and 7312 cards can be programmed to generate vectors or restart instructions as a result of an interrupt. This feature is enabled by inserting W2 7-8. Response to an interrupt acknowledge cycle can be disabled by removing W2 7-8 and inserting W2 9-10.

Interrupt Request

If the 7314A and 7312 are used in systems that implement the STD BUS serial or polled interrupt prioritization methods, jumper W2 11-12 connects the interrupt requests of both SCCs to the STD BUS INTRQ* line.

If the system interrupt prioritization scheme requires the 7314A or 7312 INTRQ* to be handled off the bus, such as a parallel interrupt prioritization scheme, disconnect INTRQ* from the STD BUS by removing W2 11-12 on the 7314A only. The two SCC interrupts can appear separately at J1 (W2 13-14 removed) or they can be wire-ORed (W2 13-14 installed).

PRIORITY CHAIN

The 7314A and 7312 can be used with the STD BUS interrupt priority chain with the STD BUS PCI and PCO pins. Priority is based on position in the card rack with the highest priority on the right. (Note that some manufacturers make card racks with highest priority on the left.) The 7314A, 7312, and other cards in a system that make use of the priority chain should be installed in the card rack according to their priority.

The 7314A and 7312 have a fixed priority for the various interrupt sources on the card but, as a card, they may be higher or lower than other cards.

Cards that use the priority chain must be placed next to each other with no unused card slots in between. Other cards that do not use the priority chain can be installed between cards using the priority chain. However, these other cards must pass on the priority chain by connecting pins 51 and 52 (priority chain out and priority chain in) together onboard. This is standard practice on Pro-Log STD BUS cards.

The cards that use the priority chain, as a group, can be placed anywhere in the card rack and other cards can be placed anywhere on either side of the group. Other cards may have empty slots in between them or may be placed in the group provided they pass on the priority chain.

The 7314A and 7312 can also be used in systems that do not use the STD BUS priority chain (such as systems with master-slave 8259 priority interrupt controllers). The INTRQ* line of the 7314A and 7312 is provided at connector J1 on the front of the card and can be disconnected from the STD BUS by removing jumper W2 11-12.

Select Input

The function of jumper W2 17-18 is dependent on the version of the 7314A and 7312 used.

With the 7314A-1 and 7312-1 versions, used with the 8088 CPU, W2 17-18 selects advanced write or normal write operation. Install W2 17-18 for advanced write operation, and remove the jumper for normal write operation.

With the 7314A-2 and 7312-2 versions, W2 17-18 selects operation with either a Z80 or 8085 CPU. Install W2 17-18 for operation in a Z80 CPU system, and remove the jumper for operation in an 8085 CPU system.

RS-232-C Interface

Each serial communications channel has an RS-232-C interface and jumpers that select whether the channel operates as data communication equipment (DCE) or data terminal equipment (DTE) (see Figure 2-8).

The connectors and jumpers assigned to each channel and the as-shipped configurations are as follows:

Channel	Connector	Jumper	As-Shipped Configuration
Channel A	J4	W3	DCE
Channel B	J2	W4	DTE
Channel C[1]	J3	W5	DCE
Channel D[1]	J5	W6	DTE

[1]7314A only

To configure a channel for DTE, install the respective channel's jumpers (W3, 4, 5, or 6) horizontally (1-3, 2-4, 5-7, 6-8, 9-11, 10-12). For DCE, install the jumpers (W3, 4, 5, or 6) vertically (1-2, 3-4, 5-6, 7-8, 9-10, 11-12). For synchronous RS-232-C operation, install jumpers (W3, 4, 5, or 6) 13-15, 14-16 for DTE, or 13-14, 15-16 for DCE. If a transmit clock must be sourced by the 7314A or 7312, (W3, 4, 5, or 6) 17-18 should be installed.

Cables for use with the RS-232-C interfaces are listed in Section 1 (Figure 1-19). Figure 2-9 shows the pinouts for connectors of the RS-232-C transition cable (transition connector pins to D connector pins).

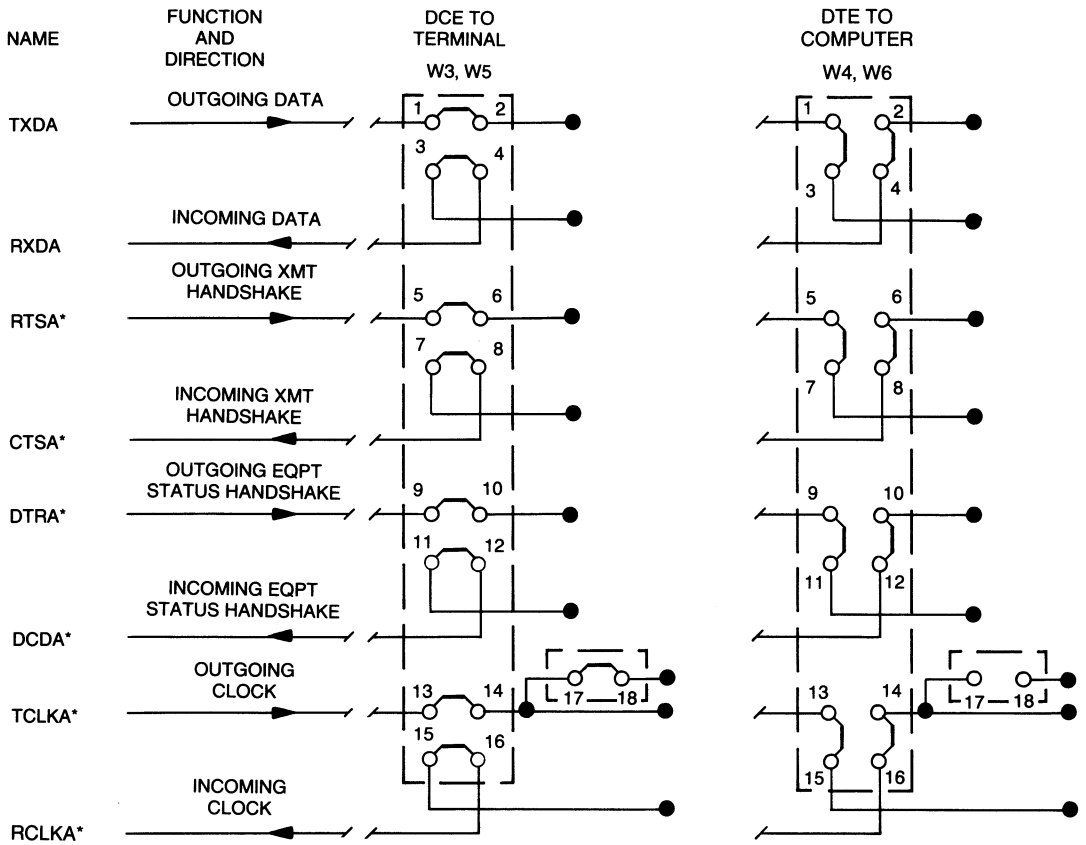


Figure 2-8. RS-232-C Serial Ports DCE/DTE Jumpers

PIN # OF SERIAL CONNECTOR J2, 3, 4, 5	SIGNAL NAME	RS-232-C PIN (D CONNECTOR)
1	N/C	1
2	N/C	14
3	TD*	2
4	RCLK (DB)	15
5	RD*	3
6	N/C	16
7	RTS	4
8	RCLK (DD)	17
9	CTS	5
10	N/C	18
11	DSR	6
12	N/C	19
13	GND	7
14	DTR	20
15	N/C	8
16	N/C	21
17	N/C	9
18	N/C	22
19	N/C	10
20	N/C	23
21	N/C	11
22	TCLK (DA)	24
23	N/C	12
24	N/C	25
25	N/C	13
26	N/C	N/C

Figure 2-9. RS-232-C Serial Port D Connector Converter

Card Configuration

Pro-Log configures jumpers W7 and W8 as appropriate for each version of the 7314A and 7312. See Figure 2-4 on page 2-3 for this as-shipped configuration.

Section 3
OPERATION AND PROGRAMMING

INTRODUCTION

This section covers the operation of the 7314A and 7312 serial ports which are controlled through I/O ports. Details of the SCC Serial Communication capabilities and internal structure, baud rate generation, interrupt options, and RS-232-C signals are included in this section.

The 7314A card contains two SCCs and four channels (A, B, C, and D). The 7312 card has one SCC with two channels (A and B). If a 7312 card is used, please disregard all references to channels C and D.

Serial Communications Capabilities

The SCC is capable of transmitting and receiving data in various formats: asynchronous, byte synchronous, and bit synchronous. Examples of each are shown in Figure 3-1. See Pro-Log's Application Note #156 (provided in the back of this manual) for more information about serial communication formats. The communication format of each RS-232-C channel is determined by the contents of the software-programmable SCC control registers.

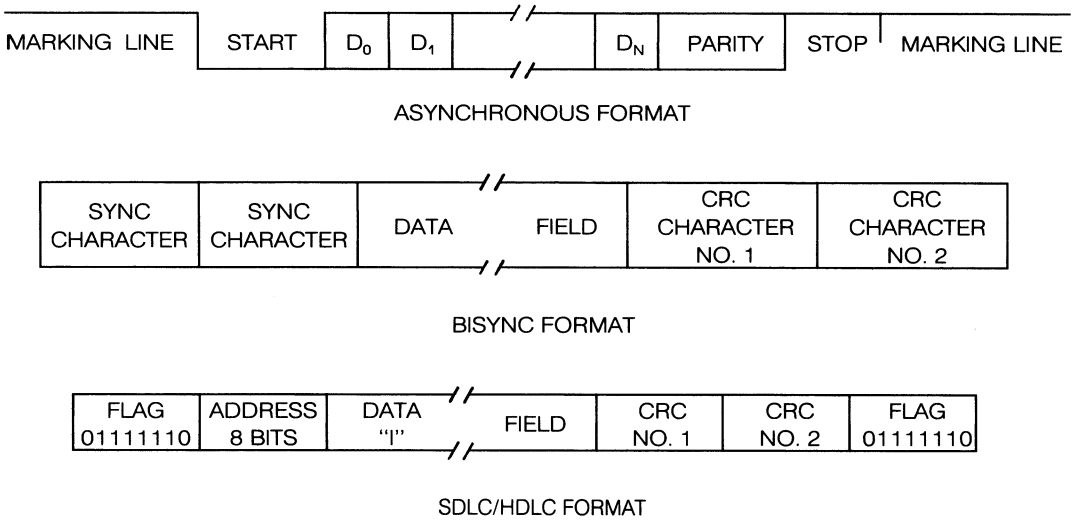


Figure 3-1. Examples of Various Serial Communication Formats

SCC Internal Structure

Operation of the serial ports depends upon the contents of the SCC internal control (write) and status (read) registers. Figure 3-2 shows the structure of the SCC registers.

Each SCC contains 16 write registers: 10 for control, 2 for sync character generation, and 2 for baud rate generation. The remaining 2 registers are shared by both channels; one is used as the interrupt vector and one as the master interrupt control. Each channel also contains eight read registers: four registers indicate status, two are used by the baud rate generator, one for interrupt vector, and one for the receive buffer. Figures 3-3 and 3-4 show the basic functions of the registers. Pro-Log presently has not tested and therefore does not support some features of the SCC. Further information concerning features of the SCC may be obtained by acquiring a copy of the Zilog Z8530 SCC technical manual.

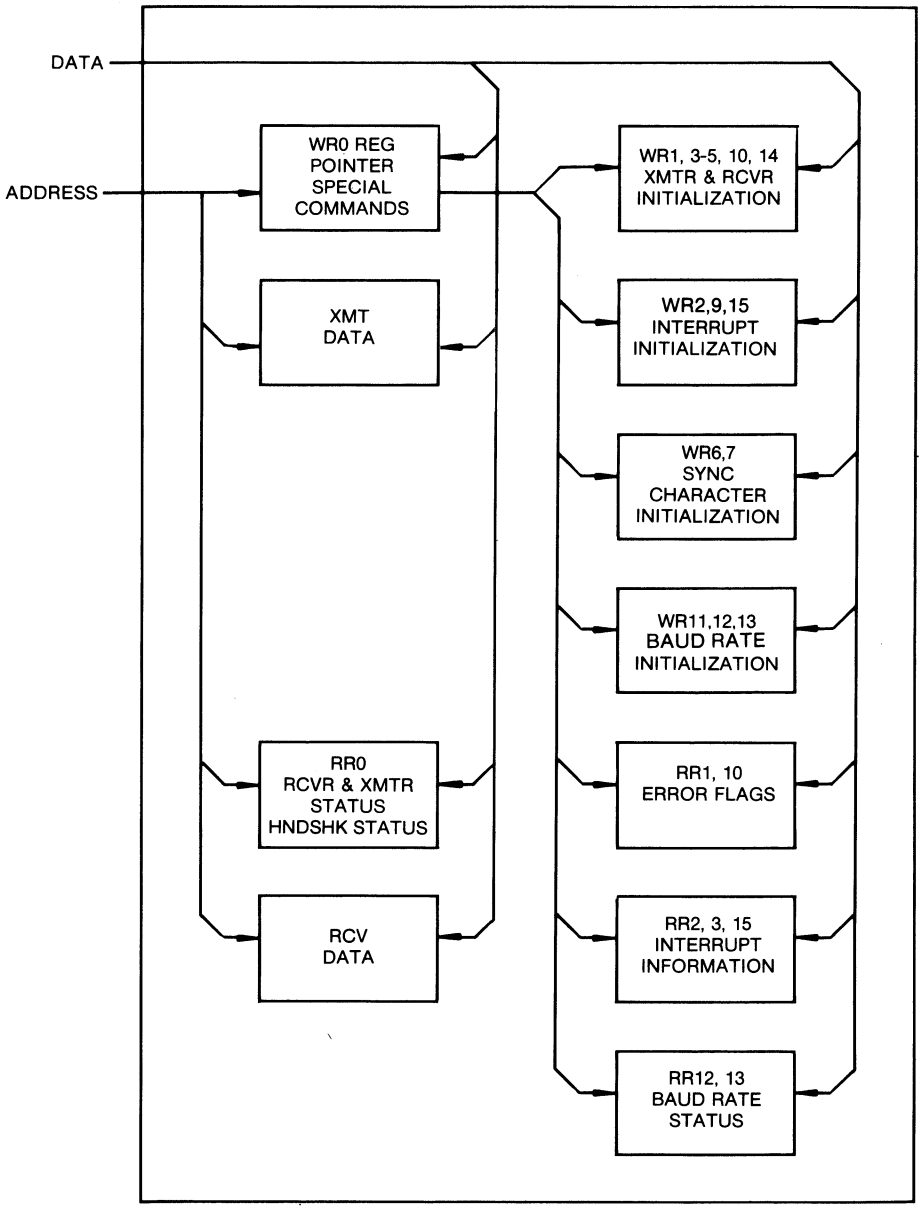


Figure 3-2. SCC Register Structure

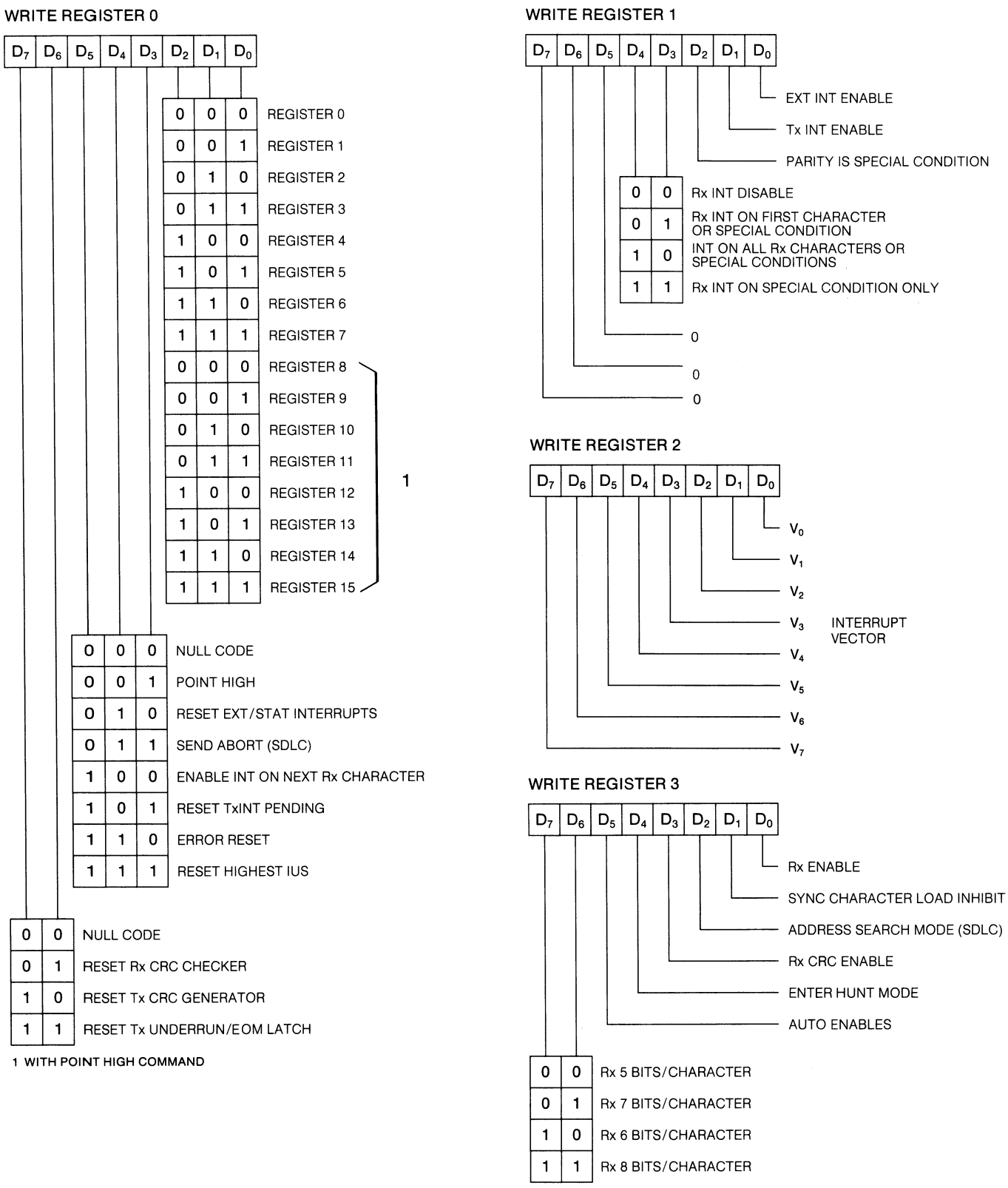
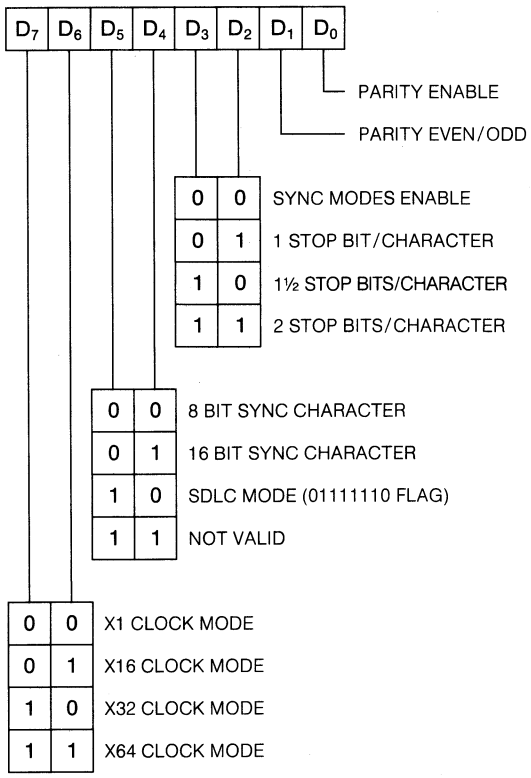
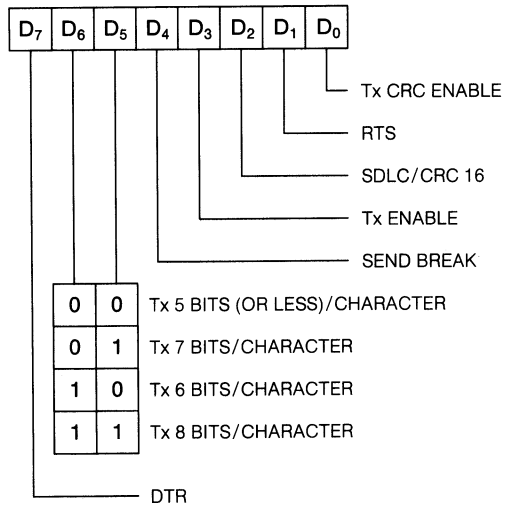

WRITE REGISTER 0
D7 D6 D5 D4 D3 D2 D1 D0
0 0 0 REGISTER 0
0 0 1 REGISTER 1
0 1 0 REGISTER 2
0 1 1 REGISTER 3
1 0 0 REGISTER 4
1 0 1 REGISTER 5
1 1 0 REGISTER 6
1 1 1 REGISTER 7
0 0 0 REGISTER 8
0 0 1 REGISTER 9
0 1 0 REGISTER 10
0 1 1 REGISTER 11
1 0 0 REGISTER 12
1 0 1 REGISTER 13
1 1 0 REGISTER 14
1 1 1 REGISTER 15
1
0 0 0 NULL CODE
0 0 1 POINT HIGH
0 1 0 RESET EXT/STAT INTERRUPTS
0 1 1 SEND ABORT (SDLC)
1 0 0 ENABLE INT ON NEXT Rx CHARACTER
1 0 1 RESET TxINT PENDING
1 1 0 ERROR RESET
1 1 1 RESET HIGHEST IUS
0 0 NULL CODE
0 1 RESET Rx CRC CHECKER
1 0 RESET Tx CRC GENERATOR
1 1 RESET Tx UNDERRUN/EOM LATCH
1 WITH POINT HIGH COMMAND
WRITE REGISTER 1
D7 D6 D5 D4 D3 D2 D1 D0
EXT INT ENABLE
Tx INT ENABLE
PARITY IS SPECIAL CONDITION
0 0 Rx INT DISABLE
0 1 Rx INT ON FIRST CHARACTER OR SPECIAL CONDITION
1 0 INT ON ALL Rx CHARACTERS OR SPECIAL CONDITIONS
1 1 Rx INT ON SPECIAL CONDITION ONLY
0
0
0
WRITE REGISTER 2
D7 D6 D5 D4 D3 D2 D1 D0
V0
V1
V2
V3
V4
V5
V6
V7
INTERRUPT VECTOR
WRITE REGISTER 3
D7 D6 D5 D4 D3 D2 D1 D0
Rx ENABLE
SYNC CHARACTER LOAD INHIBIT
ADDRESS SEARCH MODE (SDLC)
Rx CRC ENABLE
ENTER HUNT MODE
AUTO ENABLES
0 0 Rx 5 BITS/CHARACTER
0 1 Rx 7 BITS/CHARACTER
1 0 Rx 6 BITS/CHARACTER
1 1 Rx 8 BITS/CHARACTER

Figure 3-3. Write Register Bit Maps

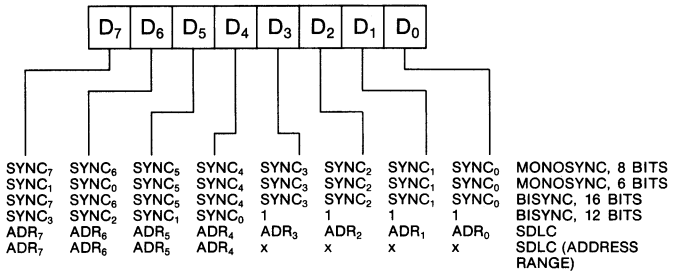
WRITE REGISTER 4


D7 D6 D5 D4 D3 D2 D1 D0
PARITY ENABLE
PARITY EVEN/ODD
0 0 SYNC MODES ENABLE
0 1 1 STOP BIT/CHARACTER
1 0 1½ STOP BITS/CHARACTER
1 1 2 STOP BITS/CHARACTER
0 0 8 BIT SYNC CHARACTER
0 1 16 BIT SYNC CHARACTER
1 0 SDLC MODE (01111110 FLAG)
1 1 NOT VALID
0 0 X1 CLOCK MODE
0 1 X16 CLOCK MODE
1 0 X32 CLOCK MODE
1 1 X64 CLOCK MODE

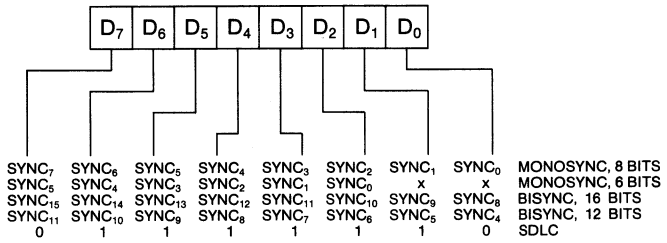
WRITE REGISTER 5



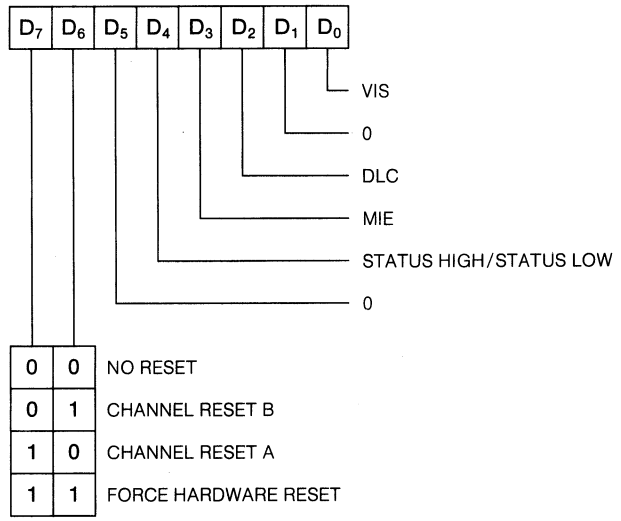
WRITE REGISTER 6


D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC1 SYNC0 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 6 BITS
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 BISYNC, 16 BITS
SYNC3 SYNC2 SYNC1 SYNC0 1 1 1 1 BISYNC, 12 BITS
ADR7 ADR6 ADR5 ADR4 ADR3 ADR2 ADR1 ADR0 SDLC
ADR7 ADR6 ADR5 ADR4 x x x x SDLC (ADDRESS RANGE)

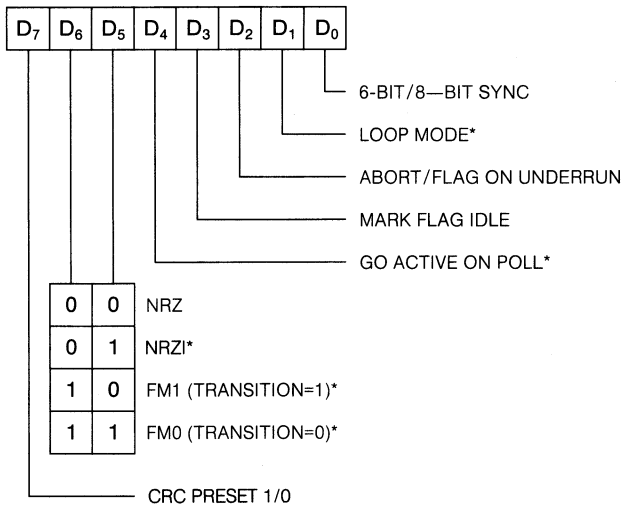
WRITE REGISTER 7


D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 x x MONOSYNC, 6 BITS
SYNC15 SYNC14 SYNC13 SYNC12 SYNC11 SYNC10 SYNC9 SYNC8 BISYNC, 16 BITS
SYNC11 SYNC10 SYNC9 SYNC8 SYNC7 SYNC6 SYNC5 SYNC4 BISYNC, 12 BITS
0 1 1 1 1 1 1 0 SDLC

WRITE REGISTER 9



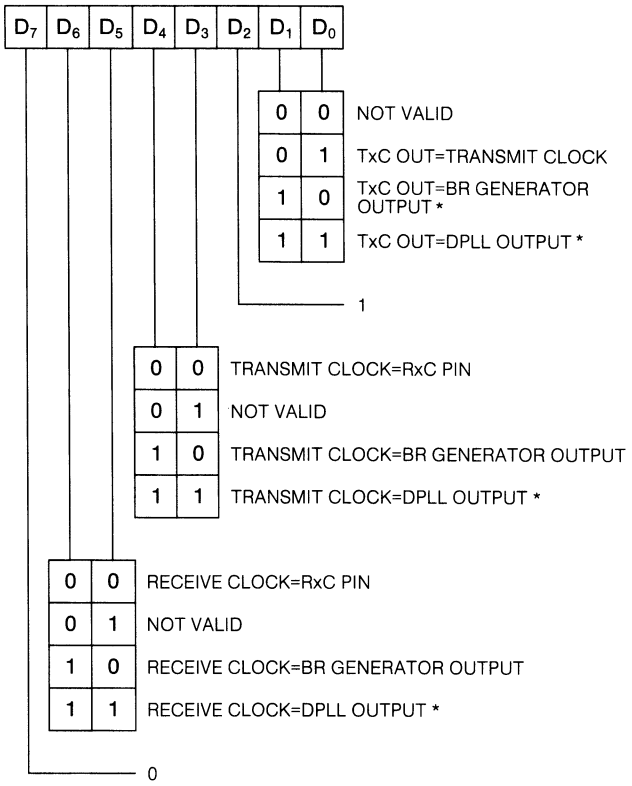
WRITE REGISTER 10



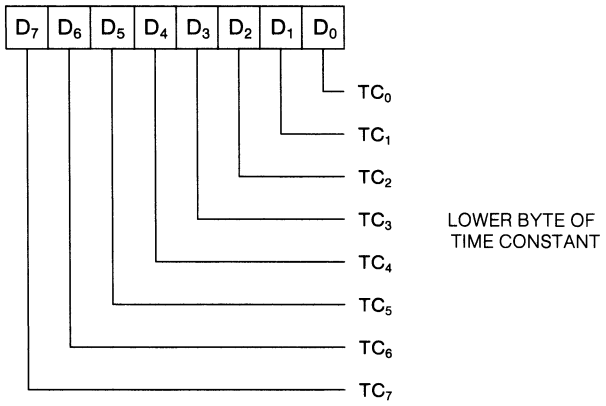
*This feature is not presently supported by Pro-Log.

Figure 3-3. (Continued)

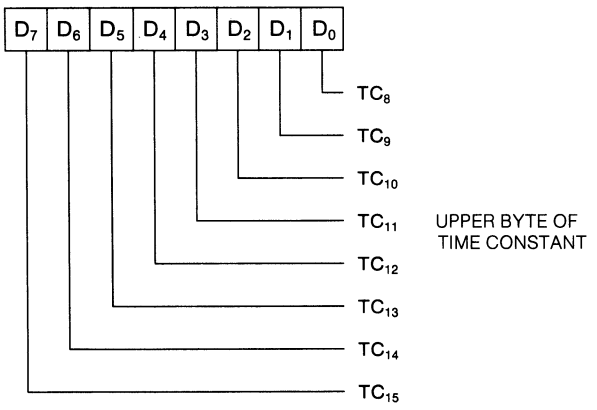
WRITE REGISTER 11


D7 D6 D5 D4 D3 D2 D1 D0
0 0 NOT VALID
0 1 TxC OUT=TRANSMIT CLOCK
1 0 TxC OUT=BR GENERATOR OUTPUT *
1 1 TxC OUT=DPLL OUTPUT *
1
0 0 TRANSMIT CLOCK=RxC PIN
0 1 NOT VALID
1 0 TRANSMIT CLOCK=BR GENERATOR OUTPUT
1 1 TRANSMIT CLOCK=DPLL OUTPUT *
0 0 RECEIVE CLOCK=RxC PIN
0 1 NOT VALID
1 0 RECEIVE CLOCK=BR GENERATOR OUTPUT
1 1 RECEIVE CLOCK=DPLL OUTPUT *
0

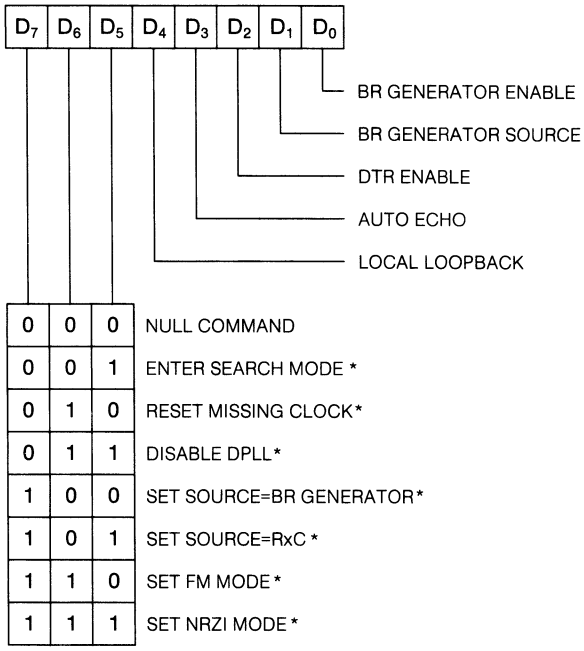
WRITE REGISTER 12



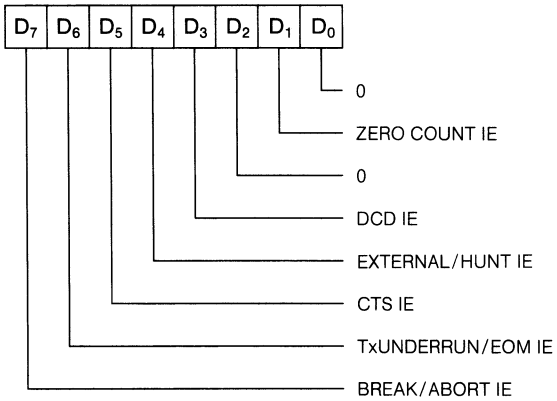
WRITE REGISTER 13



WRITE REGISTER 14


D7 D6 D5 D4 D3 D2 D1 D0
BR GENERATOR ENABLE
BR GENERATOR SOURCE
DTR ENABLE
AUTO ECHO
LOCAL LOOPBACK
0 0 0 NULL COMMAND
0 0 1 ENTER SEARCH MODE *
0 1 0 RESET MISSING CLOCK*
0 1 1 DISABLE DPLL*
1 0 0 SET SOURCE=BR GENERATOR*
1 0 1 SET SOURCE=RxC *
1 1 0 SET FM MODE *
1 1 1 SET NRZI MODE *

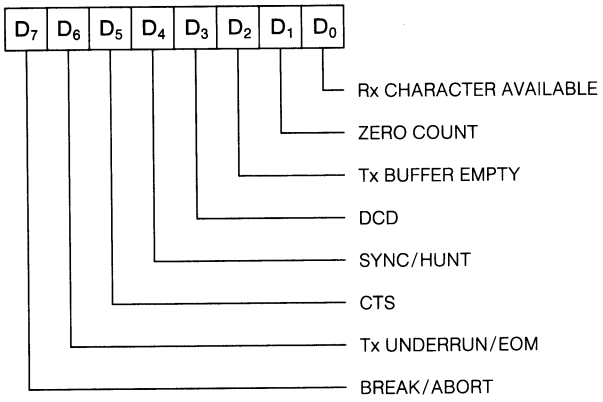
WRITE REGISTER 15



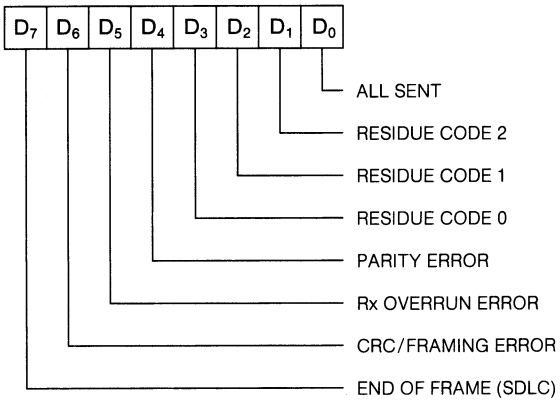
* This feature is not presently supported by Pro-Log

Figure 3-3. (Continued)

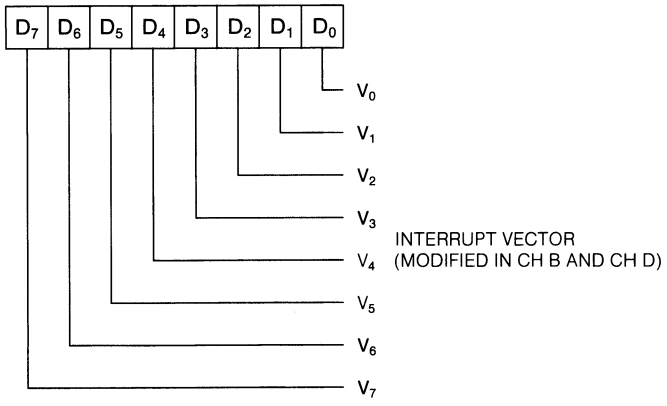
READ REGISTER 0



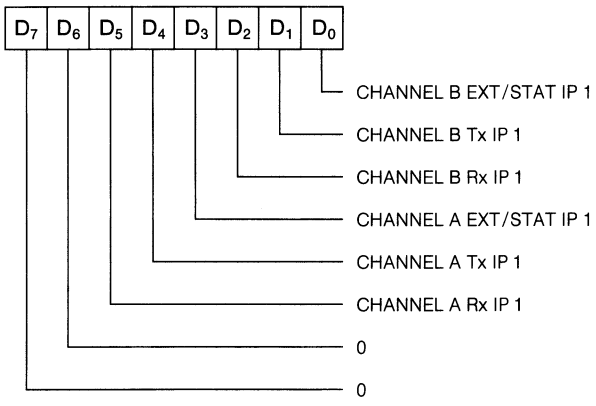
READ REGISTER 1



READ REGISTER 2

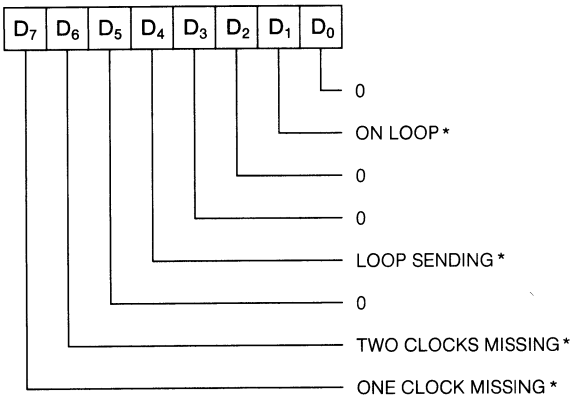


READ REGISTER 3


D7 D6 D5 D4 D3 D2 D1 D0
CHANNEL B EXT/STAT IP 1
CHANNEL B Tx IP 1
CHANNEL B Rx IP 1
CHANNEL A EXT/STAT IP 1
CHANNEL A Tx IP 1
CHANNEL A Rx IP 1
0
0

1. ALWAYS 0 IN CH B AND CH D

READ REGISTER 10



*This feature is not presently supported by Pro-Log.

READ REGISTER 12

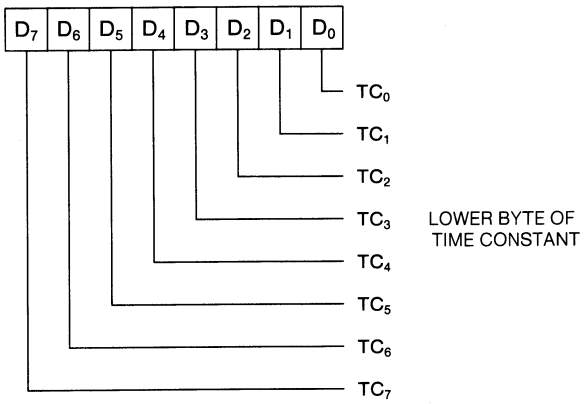
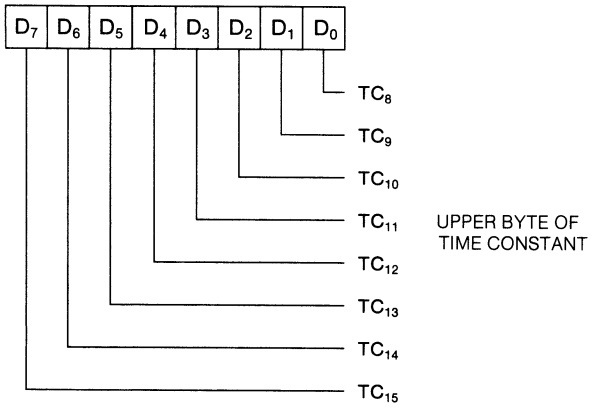


Figure 3-4. Read Register Bit Maps

READ REGISTER 13



READ REGISTER 15

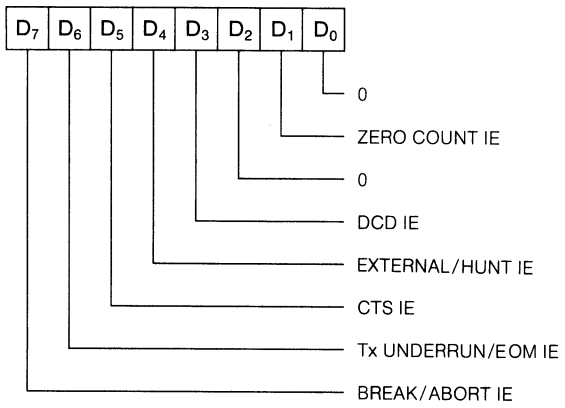


Figure 3-4. (Continued)

SCC RECEIVE AND TRANSMIT REGISTERS

The receive (RCV) data register is directly addressable and retains the data that is received over the serial interface. It holds up to three bytes in addition to the byte currently being received. The error flags in the read register are also four levels deep so the appropriate status information is available with each byte. RR1 should be read before the RCV data register as the stacks for both data and status are incremented when the data is read. If enabled, an interrupt occurs each time a character reaches the top of the stack. If there are no characters pending in the stack, a received character goes directly to the top of the stack.

If four complete bytes have been received since the last time the processor read a byte out of the RCV data register, the third byte will be overwritten. The fourth character, in the shift register, is loaded into the bottom of the stack of received characters as soon as it is completely received. If the Status Affects Vector mode is selected, an interrupt is generated with a modified vector indicating an error. This interrupt occurs when the character that overwrote the previous one reaches the top of the stack. An error flag is set in the status information associated with the fourth byte, indicating that it overwrote the previous byte.

In the Interrupt on First Received Character mode (interrupts can occur on the first character or every character), if an error occurs the data and status stacks are frozen, even if read, and prevent access to further data until an error reset is performed. Error reset is explained in the WR0 description.

The transmit (XMT) data register is directly addressable and holds the data to be transmitted. In addition to the XMT data register, there is a 20-bit shift register for the byte currently being transmitted. This allows the processor to give the SCC one byte while it is transmitting another, thereby eliminating delays between characters.

An interrupt and/or flag indicates that the XMT data register is empty and is able to accept another character. The All Sent flag in RR1 indicates that actual transmission has been completed. The All Sent condition does not generate an interrupt.

SCC CONTROL REGISTERS

All control registers require two bytes to be programmed except for WR0. The first byte contains the register select code in the least significant three bits and, if the code is for WR0, the remainder of the byte is the control word. For the other registers, the control byte is the next word. The first byte written into each channel after a reset (either with a channel reset command or hardware system reset) is directed to WR0. WR0 is shared by both channels of each SCC. While using WR0 as a pointer, care must be taken not to overwrite the pointer until after the specific desired register is written to. Programs operating in interrupt mode are susceptible to erroneous register writing.

Example:

a) WR5 is pointed to by WR0 in channel A

b) An interrupt on channel B causes WR0 to point to WR9 in channel B

c) When the software returns to channel A, WR0 remains pointing to WR9

d) The control byte that was intended for WR5 in channel A is erroneously written into WR9 of channel A

To avoid erroneously written control bytes, inhibit all interrupts while using WR0 as a pointer.

If you are using the 7314A or 7312 as the link between the PC and the STD BUS for Pro-Log's SMDS I (Input) and O (Output) commands, control bytes may be erroneously overwritten to the channel that shares the same SCC as SMDS.

For example: If channel A of the 7314A is used by SMDS, interrupts should be inhibited while using WR0 as a pointer to channel B registers. Channels C an D are not affected.

Figure 3-5 lists the functions assigned to each control register and the pages following contain detailed descriptions of these registers. All references to pin states are as seen at the SCC. The state programmed into a bit is the state that appears at the pin. The RS-232-C interface inverts the signal. In most systems, the equipment at the other end of the serial port will also have an RS-232-C inversion. Pro-Log presently has not tested and therefore does not support some features of the SCC. These are marked with an asterisk (*) in the register diagram. In the bit descriptions, bits are referred to as being "set" or "reset." "Set" refers to bit DX=1 and "reset" refers to bit DX=0.

WR0 CRC initialize, initialization commands for the various modes, register pointers

WR1 Transmit/Receive interrupt and data transfer mode definition

WR2 Interrupt vector (accessed through either channel of each SCC)

WR3 Receive parameters and control

WR4 Transmit/Receive miscellaneous parameters and modes

WR5 Transmit parameters and controls

WR6 Sync characters or SDLC address field

WR7 Sync character of SDLC flag

WR8 Transmit buffer

WR9 Master interrupt control and reset (accessed through either channel of each SCC)

WR10 Miscellaneous transmitter/receiver control bits

WR11 Clock mode control

WR12 Lower byte of baud rate generator time constant

WR13 Upper byte of baud rate generator time constant

WR14 Miscellaneous control bits

WR15 External/Status interrupt control

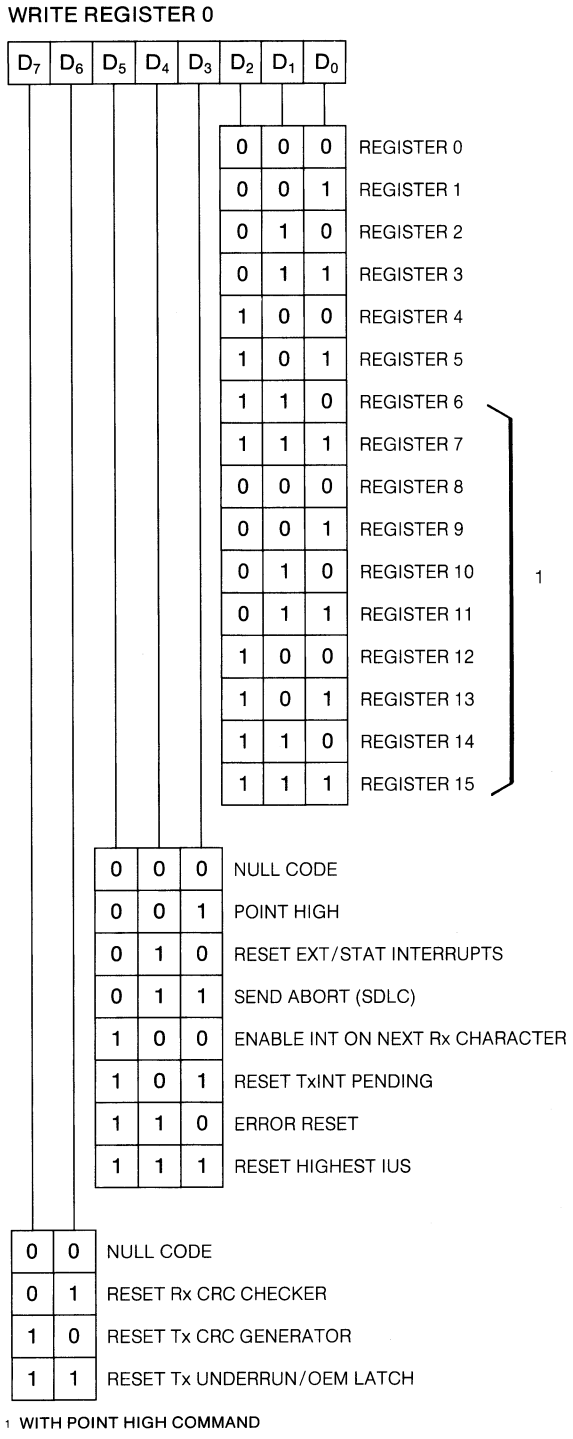

WRITE REGISTER 0
D7 D6 D5 D4 D3 D2 D1 D0
0 0 0 REGISTER 0
0 0 1 REGISTER 1
0 1 0 REGISTER 2
0 1 1 REGISTER 3
1 0 0 REGISTER 4
1 0 1 REGISTER 5
1 1 0 REGISTER 6
1 1 1 REGISTER 7
0 0 0 REGISTER 8
0 0 1 REGISTER 9
0 1 0 REGISTER 10
0 1 1 REGISTER 11
1 0 0 REGISTER 12
1 0 1 REGISTER 13
1 1 0 REGISTER 14
1 1 1 REGISTER 15
1
0 0 0 NULL CODE
0 0 1 POINT HIGH
0 1 0 RESET EXT/STAT INTERRUPTS
0 1 1 SEND ABORT (SDLC)
1 0 0 ENABLE INT ON NEXT Rx CHARACTER
1 0 1 RESET TxINT PENDING
1 1 0 ERROR RESET
1 1 1 RESET HIGHEST IUS
0 0 NULL CODE
0 1 RESET Rx CRC CHECKER
1 0 RESET Tx CRC GENERATOR
1 1 RESET Tx UNDERRUN/OEM LATCH
1 WITH POINT HIGH COMMAND

Figure 3-5. SCC Control Register Functions

D7-D6 **Reset Bits**

These bits specify three reset modes: reset receive CRC check, reset transmit CRC generator, and reset sending transmit underrun/ end of message latch.

D5-D4-D3 **Command Bits**

0 0 0 This is a "no operation" command that can be issued while setting the register pointer when no other effect is desired.

0 0 1 Write Registers WR8 through WR15 are accessed with this command.

0 1 0 After the status bits of RR0 have been latched by a change of one of the external/status bits, this command reenables the latch of RR0. If an external/status interrupt occurs, command 2 is issued to reenable RR0, which allows external/status interrupts to occur. The latching allows capture of a short pulse on the input pins and holds the pulse until the CPU can read it.

0 1 1 A sequence of 8 to 13 ones is generated. It is used only with the SDLC mode.

1 0 0 If the Interrupt Only on First Received Character mode is selected (D4, D3, WR1), it must be reactived after each complete message is received.

1 0 1 The transmitter interrupts when it becomes empty if transmit interrupt was enabled at D1 of WR1. This command resets the transmitter interrupt pending until after the next character has been loaded into the transmitter. This is used during periods when no characters are to be sent.

1 1 0 Parity and overrun errors are latched in RR1 until reset by this command.

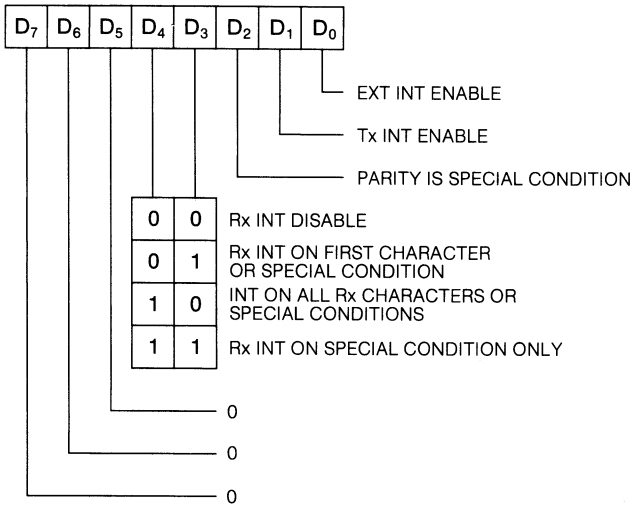
1 1 1 This command resets the highest priority Interrupt Under Service (IUS) bit and allows lower priority conditions to request an interrupt. This should be the last operation in an interrupt service routine.

D2-D0 **Register Pointers**

These bits indicate to which register the next write command is written. The first byte written into each channel after a reset (either with a channel reset command or hardware system reset) is directed to WR0 and the byte following a read or write to any register.

Write Register 1

This register controls interrupt conditions.


D7 D6 D5 D4 D3 D2 D1 D0
EXT INT ENABLE
Tx INT ENABLE
PARITY IS SPECIAL CONDITION
0 0 Rx INT DISABLE
0 1 Rx INT ON FIRST CHARACTER OR SPECIAL CONDITION
1 0 INT ON ALL Rx CHARACTERS OR SPECIAL CONDITIONS
1 1 Rx INT ON SPECIAL CONDITION ONLY
0
0
0

D7-D5 The 7314A and 7312 requires that these bits be reset.

D4-D3 **Receiver Interrupt Mode**

These bits specify the character available interrupt conditions. If the Vector Includes Status (VIS bit, D0, WR9) is enabled, the specific vector generated depends upon whether or not an error occurs. The special receive condition interrupt occurs if there is a receiver error, otherwise the receive character available interrupt occurs.

D2 **Parity is special condition**

If D2 is set, a parity error results in a special condition and modifies the interrupt vector. If parity is disabled (D0, WR4), this bit is ignored.

D1 **Transmitter Interrupt Enable**

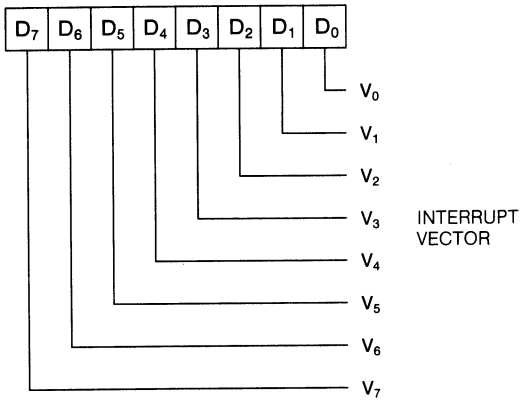
This bit allows interrupts whenever the transmitter buffer becomes empty. This interrupt generates the Transmit Buffer Empty vector of RR2 if the Vector Includes Status (VIS) bit is set (D0, WR9).

D0 **External Interrupt Enable**

This bit allows interrupts to occur as a result of transitions on this DCD*, CTS*, or SYNC* (external interrupt) lines, or as a result of a break condition (read at RR0), or the beginning of sending CRC or sync characters. These external/status change vector interrupts modify the vector of RR2 if the VIS bit is programmed (D0, WR9).

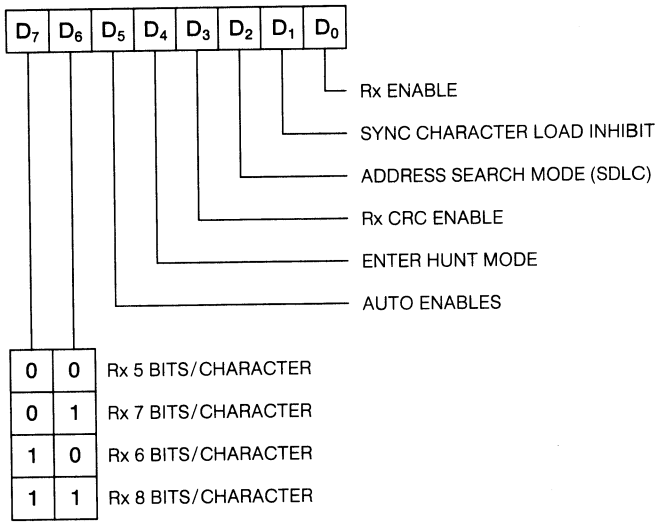
Write Register 2

This is the interrupt vector register. Only one vector register exists in each SCC. It may be programmed with a restart instruction for the 8085 or a unique vector for the Z80 and 8088. Some of the bits may be programmed to change according to SCC status.



Write Register 3

This register contains receive control parameters, the number of bits per character, and the receive enable.


D7 D6 D5 D4 D3 D2 D1 D0
Rx ENABLE
SYNC CHARACTER LOAD INHIBIT
ADDRESS SEARCH MODE (SDLC)
Rx CRC ENABLE
ENTER HUNT MODE
AUTO ENABLES
0 0 Rx 5 BITS/CHARACTER
0 1 Rx 7 BITS/CHARACTER
1 0 Rx 6 BITS/CHARACTER
1 1 Rx 8 BITS/CHARACTER

D7-D6 **Receiver Bits/Character**

These bits determine the number of serial receive bits that are assembled to form a character. An additional parity bit is received if parity is enabled at D0 of WR4.

D5 **Auto Enables**

If this mode is selected, the DCD* and CTS* lines for the channel are receiver and transmitter enables, respectively. If the mode is not selected, the DCD* and CTS* are inputs to bits in RR0. Transitions on these lines can cause an interrupt if external interrupts are enabled at D0 of WR1.

D4 **Enter Hunt Mode**

If character synchronization is lost or, if in the SDLC mode, the incoming message is not needed, the Hunt mode may be re-entered by setting this bit.

D3 **Receiver CRC Enable**

When this bit is set, CRC calculation begins at the start of the last character transferred from the receive register to the buffer stack. This occurs regardless of the number of characters in the stack.

D2 **Address Search Mode**

When this bit is set and the SDLC mode is selected, no interrupts are generated unless an address match (programmed or global) occurs.

D1 **Sync Character Load Inhibit**

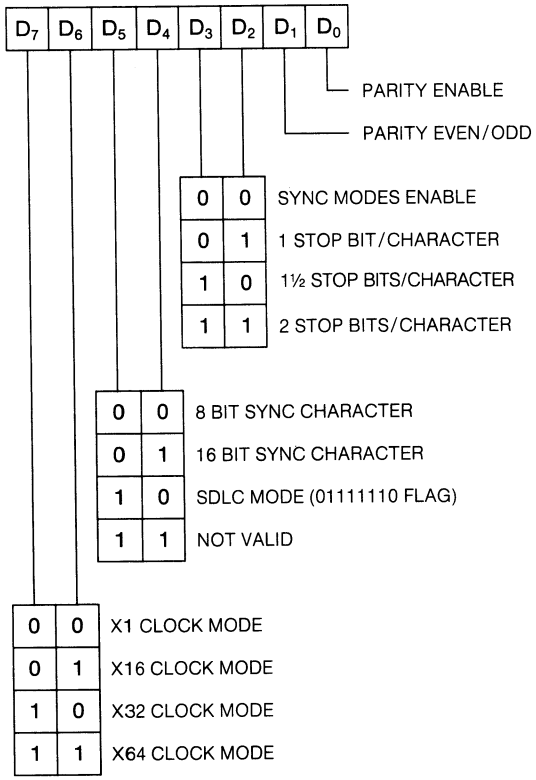
When this bit is set, sync characters preceding a message are not loaded into the receive buffers. The CRC calculation is not terminated if no sync characters are generated.

D0 **Receiver Enable**

This bit enables the receiver and is normally left set throughout operation. If the receiver is disabled while receiving a character, the character is lost and no interrupt occurs.

Write Register 4

WR4 contains transmit and receive parameters and modes, clock rate, number of sync characters, stop bits, and parity.


D7 D6 D5 D4 D3 D2 D1 D0
PARITY ENABLE
PARITY EVEN/ODD
0 0 SYNC MODES ENABLE
0 1 1 STOP BIT/CHARACTER
1 0 1½ STOP BITS/CHARACTER
1 1 2 STOP BITS/CHARACTER
0 0 8 BIT SYNC CHARACTER
0 1 16 BIT SYNC CHARACTER
1 0 SDLC MODE (01111110 FLAG)
1 1 NOT VALID
0 0 X1 CLOCK MODE
0 1 X16 CLOCK MODE
1 0 X32 CLOCK MODE
1 1 X64 CLOCK MODE

D7-D6 **Clock Rate**

The clock rate specifies the multiplier between clock and data rates. The same multiplier is used for both the receiver and transmitter. In all modes, the system clock must be at least five times the data rate. If the X1 clock rate is selected, bit synchronization must be accomplished externally. In synchronous modes, the X1 clock mode is forced.

D5-D4 **Sync Modes**

These bits select character synchronization options.

D3-D2 **Stop Bits**

These bits determine the number of stop bits added to each character sent. The receiver always checks for one stop bit.

D1 **Parity Even/Odd**

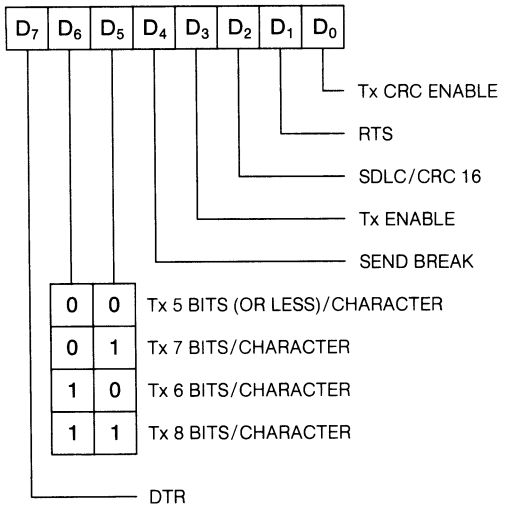
If parity is specified by bit D0, this bit determines whether the parity is sent or checked as even (D1=1) or odd (D1=0).

D0 **Parity**

When this bit is set, an additional bit position (in addition to those specified in the bits/character control of WR3 or 4) is added to transmitted data and is expected in receive data.

Write Register 5

WR5 contains transmit parameters and controls, number of transmit bits per character, and the transmit CRC enable.



D7 **Data Terminal Ready**

This is the control bit for the DTR* output pin for the channel. When set, DTR* is active low. When reset, DTR* is inactive high.

D6-D5 **Transmit Bits/Character**

These bits determine the number of bits that are sent from each byte transferred to the transmit buffer. An additional parity bit is sent if parity is enabled at D0 of WR4. Bits are assumed to be right justified. If the 5 bits/character mode is selected, the unused bits of each data byte must be zero.

D4 **Send Break**

When set, this bit forces the TxD pin for the channel to spacing low, regardless of any data being transmitted. When reset, the TxD pin for the channel is released.

D3 **Transmit Enable**

This bit enables the transmitter. Data is not transmitted and the transmit data pin remains high until the transmitter is enabled. If the transmitter is disabled while

sending, the character being sent will be completed. An interrupt still occurs when the XMT Data register becomes empty. Only the character being sent is completed. If one is pending in the XMT Data register, it remains pending. CRC characters are not sent if the transmitter is disabled during CRC transmission.

D2 **SDLC/CRC**

This bit selects the CRC code used by the transmitter and receiver. When the bit is set, the CRC-16 polynomial $X^{16} + X^{15} + X^2 + X^1$ is used and the CRC registers are reset to 0. When D2 is reset, the SDLC polynomial $X^{16} + X^{12} + X^2 + 1$ is used.

D1 **Request to Send**

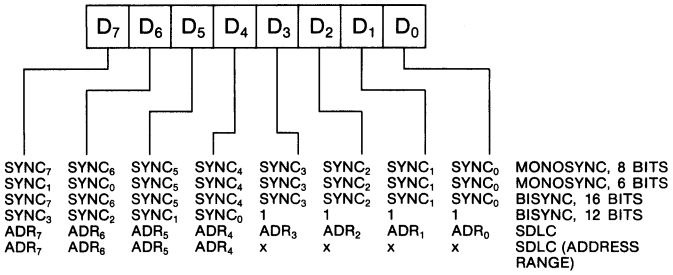
This is the control bit for the RTS* output pin for the channel. When D1 is set, the RTS* line goes active low. When D1 is reset, the RTS* pin is inactive high only after the transmitter is empty.

D0 **Transmit CRC Enable**

If this bit is set at the time of loading the character from the transmit buffer to the transmit shift register, the CRC is calculated on the character. The CRC is not automatically sent when the transmitter is empty unless this bit is set.

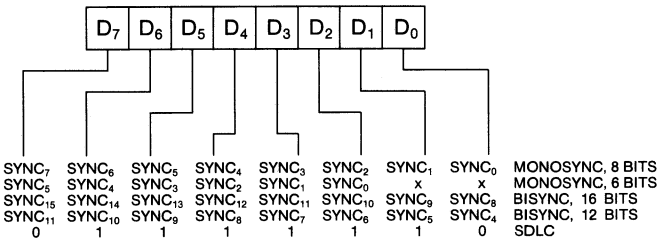
Write Register 6

WR6 contains the first 8 bits of a BISYNC sequence. It must contain the check address (if used) for SDLC mode and the sync character in the 8-bit sync mode.


D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC1 SYNC0 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 6 BITS
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 BISYNC, 16 BITS
SYNC3 SYNC2 SYNC1 SYNC0 1 1 1 1 BISYNC, 12 BITS
ADR7 ADR6 ADR5 ADR4 ADR3 ADR2 ADR1 ADR0 SDLC
ADR7 ADR6 ADR5 ADR4 x x x x SDLC (ADDRESS RANGE)

Write Register 7

WR7 contains an 8-bit sync character of a 16-bit synchronization sequence, or the SDLC flag 01111110.

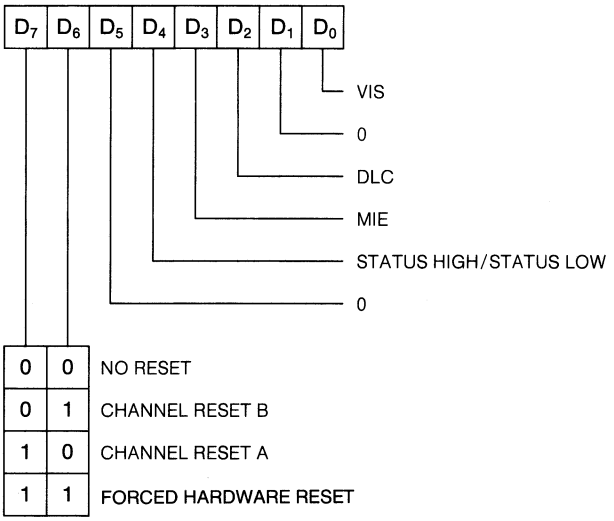

D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 x x MONOSYNC, 6 BITS
SYNC15 SYNC14 SYNC13 SYNC12 SYNC11 SYNC10 SYNC9 SYNC8 BISYNC, 16 BITS
SYNC11 SYNC10 SYNC9 SYNC8 SYNC7 SYNC6 SYNC5 SYNC4 BISYNC, 12 BITS
0 1 1 1 1 1 1 0 SDLC

Write Register 8

WR8 is the transmit buffer.

Write Register 9

WR9 contains the master interrupt control and reset. Both channels of each SCC share WR9 and it can be accessed through either channel. The interrupt control bits can be programmed at the same time as the reset command.



D7-D6 **Reset Bits**

Setting either D7 or D6 disables the receiver and transmitter, forces the handshake signals high, resets the interrupt pending (IP) and interrupt under service (IUS) bits, and disables all interrupts in the corresponding channel. Four additional PCLK cycles (814 nanoseconds) must be allowed before additional command or control bytes are written to each SCC.

NO RESET. This is a "no operation" command that is used when WR9 is written to but no resets are implemented.

FORCED HARDWARE RESET: This command has the same effect as a hardware reset except the MIE, status high/status low, and DLC bits are reset.

D5 This bit is not used and must be reset.

D4 **Status High/Status Low**

When D4 is set, bits V6, V5, and V4 of the interrupt vector will be modified. When D4 is reset, bits V3, V2, and V1 will be modified (see Figure 3-7).

D3 **Master Interrupt Enable**

SCC interrupts are enabled when this bit is set. When D3 is reset, SCC interrupts are disabled.

D2 **Disable Lower Chain**

When this bit is set, the IEO pin of the selected SCC is forced low, disabling lower-priority devices on the daisy chain from requesting interrupts.

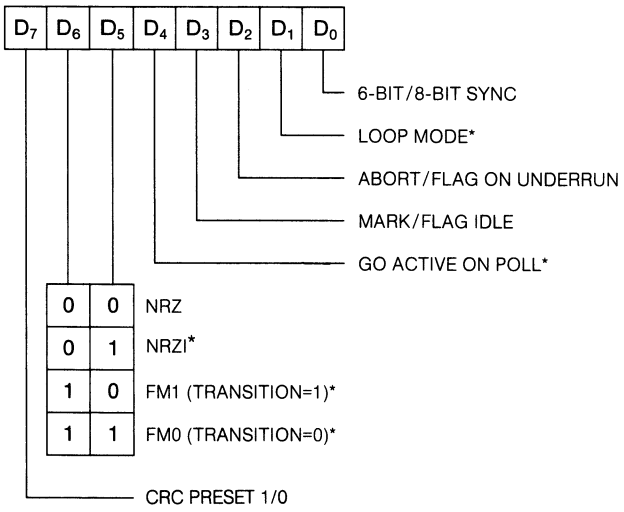
D1 This bit is not used on the 7314A and 7312 and must be reset.

D0 **Vector Includes Status**

If this bit is set, the vector returned by the SCC in response to an interrupt acknowledge cycle is variable, according to Figure 3-7. If D0 is reset, the vector returned is that programmed in WR2.

Write Register 10

WR10 contains miscellaneous transmitter and receiver control bits, NRZI, NRZ, and FM encoding options, and the CRC reset.



*This feature is not presently supported by Pro-Log.

D7 **CRC Preset I/O**

The initialized condition of the received CRC checker and the transmit CRC generator are selected by this bit. If D7 is set, the CRC generator and checker are preset to 1. If D7 is reset, the CRC generator and checker are preset to 0. Either option can be used with either CRC polynomial. This bit is ignored in asynchronous modes.

D6-D5 **Data Encoding**

These bits control the clock encoding method used for both the transmitter and receiver of the selected SCC channel. Only NRZ clock encoding (D6D5=00) is presently supported by Pro-Log.

D4 **Go Active on Poll**

This is used for SDLC loop mode operation which is not presently supported by Pro-Log.

D3 **Mark/Flag Idle**

This bit affects SDLC idle operation. With D3 set, the transmitter sends continuous ones after the closing flag of a frame. If D3 is reset, continuous flags are sent during idle operation.

D2 **Abort/Flag on Underrun**

In SDLC operation, if D2 is set, the SCC sends an abort and a flag instead of a CRC on a transmit underrun. If D2 is reset, a CRC is sent on a transmit underrun. If both D3 and D2 are set, all ones are transmitted after the transmit underrun. This bit should be set after the first byte of data is sent to the SCC and reset immediately after the last byte of data, so the SDLC frame ends with a CRC and a flag.

D1 **Loop Mode**

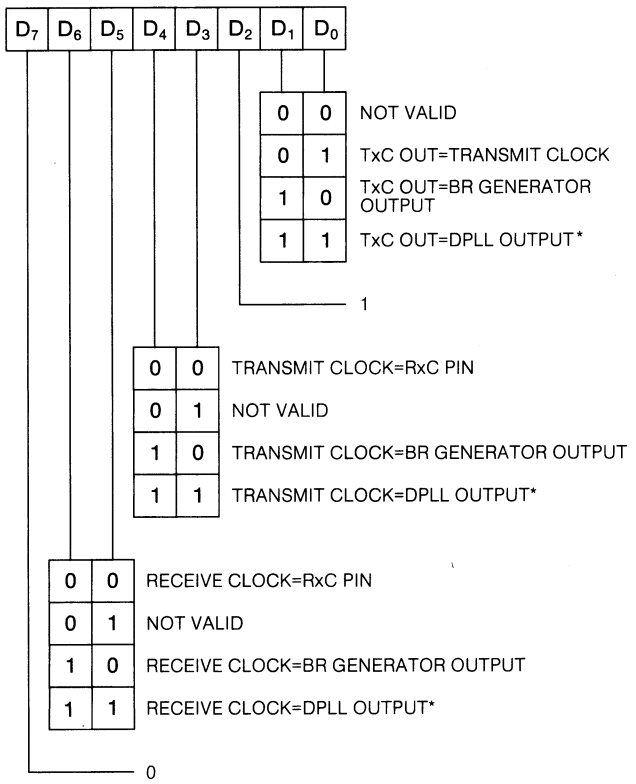
This bit is used for SDLC loop mode operation which is not presently supported by Pro-Log.

D0 **6-Bit/8-Bit Sync**

This bit selects a synchronous mode. In monosync mode, if D0 is set, the receiver and transmitter sync characters are six bits long instead of eight bits. In BISYNC mode, if D0 is set the receiver sync character is 12 bits long and the the transmitter sync character remains 16 bits long. This bit is ignored in asynchronous and SDLC modes.

Write Register 11

WR11 contains the clock mode control and directs the source of the transmit and receive clocks.


D7 D6 D5 D4 D3 D2 D1 D0
0 0 NOT VALID
0 1 TxC OUT=TRANSMIT CLOCK
1 0 TxC OUT=BR GENERATOR OUTPUT
1 1 TxC OUT=DPLL OUTPUT*
1
0 0 TRANSMIT CLOCK=RxC PIN
0 1 NOT VALID
1 0 TRANSMIT CLOCK=BR GENERATOR OUTPUT
1 1 TRANSMIT CLOCK=DPLL OUTPUT*
0 0 RECEIVE CLOCK=RxC PIN
0 1 NOT VALID
1 0 RECEIVE CLOCK=BR GENERATOR OUTPUT
1 1 RECEIVE CLOCK=DPLL OUTPUT*
0

*This feature is not presently supported by Pro-Log.

D7 This bit must be reset for operation with the 7314A and 7312.

D6-D5 **Receive Clock**

These bits determine the source of the receive clock. For normal asynchronous operations, the receive clock = baud rate generator output (D6D5=10). For normal synchronous operation, the receive clock = RxC pin (D6D5=00). Pro-Log does not presently support the receive clock = BR generator output (D6D5=11) feature on the 7314A and 7312.

D4-D3 These bits determine the source of the transmit clock. For normal operation, D4D3=10. The transmit clock can also be sourced by the received clock (D4D3=00). D4D3=01 is not valid. Pro-Log does not presently support the D4D3=11 feature on the 7314A and 7312.

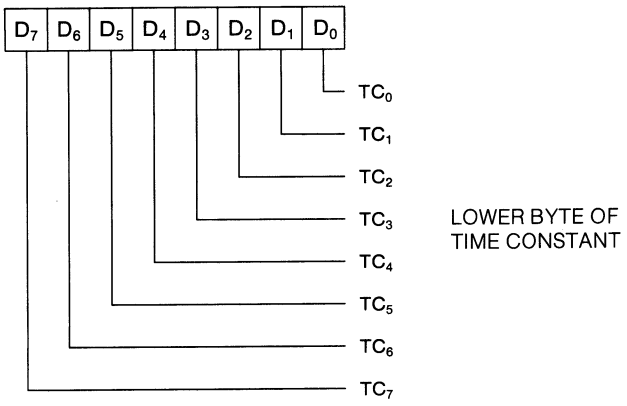
D2 This bit must be set for operation with the 7314A and 7312.

D1-D0 **Transmit Clock Output Source**

These bits determine the signal output at the SCC transmit clock pins. For normal synchronous modes, D1D0=01. If the transmit clock is sourced by the RxC input, D1D0=01. D1D0=00 is not valid. Pro-Log does not presently support the D1D0=11 feature on the 7314A and 7312.

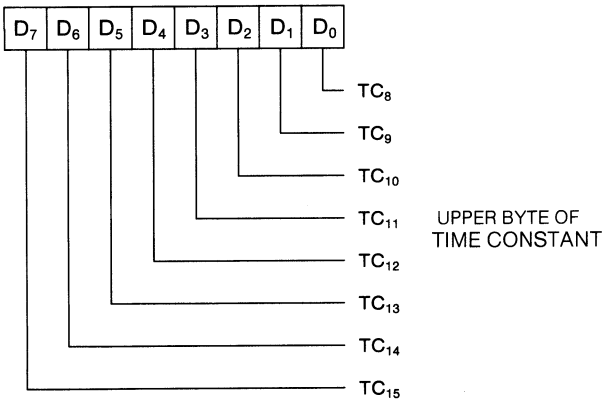
Write Register 12

WR12 contains the lower byte of the baud rate generator time constant. See Figure 3-9, page 3-22 for typical baud rate hex values.



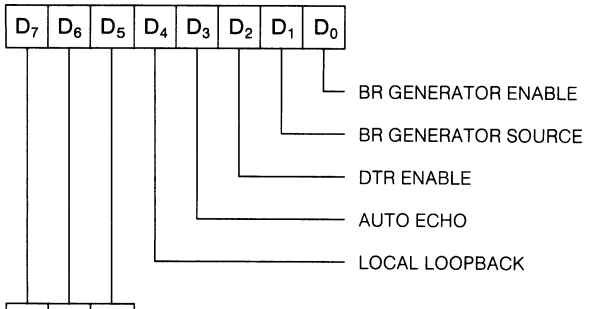
Write Register 13

WR13 contains the upper byte of the baud rate generator time constant. See Figure 3-9, page 3-22 for typical baud rate hex values.



Write Register 14

WR14 contains miscellaneous control bits: baud rate generator, phase-locked loop control, auto echo, and local loopback modes.



0	0	0	NULL COMMAND
0	0	1	ENTER SEARCH MODE*
0	1	0	RESET MISSING CLOCK*
0	1	1	DISABLE DPLL*
1	0	0	SET SOURCE=BR GENERATOR*
1	0	1	SET SOURCE=RxC*
1	1	0	SET FM MODE*
1	1	1	SET NRZI MODE*

*This feature is not presently supported by Pro-Log.

D7-D5 **Digital Phase-Locked Loop Command Bits**

These bits encode the eight commands for the digital phase-locked loop (DPLL). Pro-Log does not presently support these features on the 7314A and 7312.

NULL CODE: This command has no effect on the DPLL.

ENTER SEARCH MODE: This command resets the clock missing latches in RR10 and arms the latches to detect the next missing clock edge. This is only used in the FM mode.

DISABLE DPLL: This command disables the DPLL, resets the clock missing latches in RR10, and forces a Search mode state.

SET SOURCE = RXC: This command forces the DPLL clock to come from the RxC pin.

SET FM MODE: This command directs the DPLL to recover the clock from FM or Manchester-encoded data.

SET NRZI MODE: This command directs the DPLL to operate in NRZI mode.

D4 **Local Loopback**

When this bit is set, the transmitter output is internally connected to the receiver input. Whenever a byte is transmitted out, it is also routed to the receive buffer. The receiver ignores external data at the RxD input and CTS and DCD inputs are disabled. A change in CTS or DCD can still generate an interrupt.

D3 **Auto Echo**

When this bit is set, data is received at the RxD input and echoed out the transmit data pin. Only data received at the RxD input is transmitted at the TxD output. CTS is disabled as a transmit enable, but DCD can still be utilized as the receiver enable.

D2 **DTR Enable**

This bit should be reset to zero for the DTR pin to follow the state of the DTR bit in WR5.

D1 **Baud Rate Generator Source**

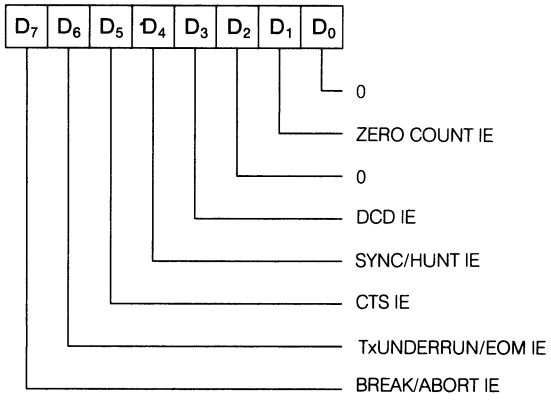
This bit selects the source of the clock for the baud rate generator. If this bit is set, the clock is the SCC's 4.9152 MHz PCLK input. If D1 is reset, the baud rate generator source is the receive clock input.

D0 **Baud Rate Generator Enable**

When this bit is set, the baud rate generator is enabled. With D0 reset, the baud rate generator is disabled.

Write Register 15

WR15 contains external/status interrupt response control.



D7 **Break/Abort Interrupt Enable**

When this bit is set, a change in the break/abort status of the receiver generates an external/status interrupt.

D6 **Transmit Underrun/End of Message**

When D6 is set, a change in the Tx Underrun/EOM latch in the receiver causes an external/status interrupt.

D5 **CTS Interrupt Enable**

When D5 is set, a change in the CTS pin causes an external/status interrupt.

D4 **External/Hunt Interrupt Enable**

In asynchronous modes when D4 is set, an external interrupt at the user interrupt connector (J1) causes an external/status interrupt. In synchronous modes if D4 is set, an external/status interrupt is generated when the hunt bit in the receiver changes state.

D3 **DCD Interrupt Enable**

When D3 is set, a change in the DCD pin causes an external/status interrupt.

D2 This bit is not used and must be reset.

D1 **Zero Count Interrupt Enable**

When D1 is set, an external/status interrupt is generated whenever the counter in the baud rate generator reaches zero.

D0 This bit is not used and must be reset.

SCC STATUS REGISTERS

All SCC read registers require two bytes to be read except RR0. The first byte must be a write to WR0 with the register select code in the least significant four bits. The next byte is read from the selected read register. The byte following a read or write to any register is directed to register 0. Figure 3-6 lists the functions assigned to each status register. A detailed description of each status register is contained on the following pages.

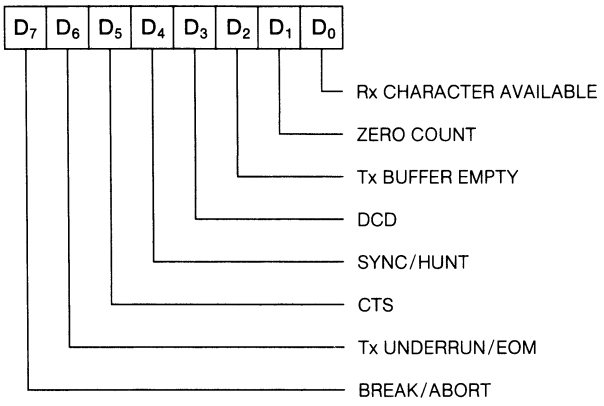
All references to pin states are as seen at the SCC. The state that appears at the pin is the state that is seen in the status bit. The RS-232-C interface inverts the signal. In most RS-232-C systems, the equipment at the other end of the serial port also has an RS-232-C inversion. Pro-Log presently does not support some features of the SCC. These are marked with an asterisk (*) in the register diagram. In the bit descriptions, bits are referred to as being "set" or "reset." "Set" refers to bit DX=1 and "reset" refers to bit DX=0. All control and status registers apply to both SCCs.

RR0	Transmit/Receive buffer status and External status
RR1	Special Receive Condition status
RR2	Unmodified interrupt vector (channels A and C only) Modified interrupt vector (channels B and D only)
RR3	Interrupt Pending bits (channels A and C only)
RR8	Receive buffer
RR10	Miscellaneous status
RR12	Lower byte of baud rate generator time constant
RR13	Upper byte of baud rate generator time constant
RR15	External/Status interrupt information

Figure 3-6. Read Register Functions

Read Register 0

RR0 contains transmit and receive buffer status and external status.



D7 **Break/Abort**

In asynchronous modes, this bit is set when a "break" (continuous zeros) is detected. After the Reset external/status interrupts command (command 2, WR0) has been executed, the break bit is reset when the break stops. If external status interrupts have been enabled (D0, WR1), both the detection and termination of the break sequence causes an interrupt. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9). In SDLC mode, an abort sequence (7 or more ones) sets this bit. It is not used in other synchronous modes.

D6 **Transmit Underrun/End of Message**

When this bit has been reset by the "reset Tx Underrun/EOM Latch" command in WR0 and a transmit underrun occurs, an external/status interrupt is generated. Only a zero-to-one transition of this bit causes an interrupt. This bit is significant only when operating in synchronous modes.

D5 **Clear to Send**

This bit shows the status of the channel CTS* pin at the time of the first change of any of the five external/status bits (DCD, CTS, Sync/Hunt, Break/Abort, or Transmit Underrun/EOM) after power-on, channel reset, or an external/status interrupt (command 2). Odd numbers of transitions on the CTS pin while another external/status interrupt is pending also causes an external/status interrupt condition. To get the current state of the CTS*

pin, this bit must be read immediately following a reset external/status interrupt command (command 2, WR0). A transition at the CTS* pin causes an interrupt if external interrupts are enabled at D0 of WR1. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9).

D4 **Sync/Hunt**

In synchronous modes, this bit is reset when character synchronization is achieved and is set by writing the Enter Hunt Mode Bit. In asynchronous modes, this bit is identical in operation to the CTS bit, except it reports the state of the sync (external interrupt) pin.

D3 **Data Carrier Detect**

This bit shows the status of the DCD* pin at the time of the first change of any of the five external/status bits (DCD, CTS, Sync/Hunt, External Interrupt, Break/Abort, or Transmit Underrun/EOM) after power-on, channel reset, or an external/status interrupt (command 2). The change of some of these bits latches the status bits of RR0. To get the current state of the DCD* pin, this bit must be read immediately following a reset external/status interrupt command (command 2, WR0). Any odd number of DCD transitions while another external/status interrupt is pending also causes an external/status interrupt condition. A transition at the DCD* pin causes an interrupt if external interrupts are enabled at D0 of WR1. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9).

D2 **Transmit Buffer Empty**

This bit is set whenever the channel transmit buffer is empty except during CRC transmission in synchronous mode. This causes an interrupt that modifies the vector of RR2 if the Status Affects Vector bit is programmed (D0, WR9).

D1 **Zero Count**

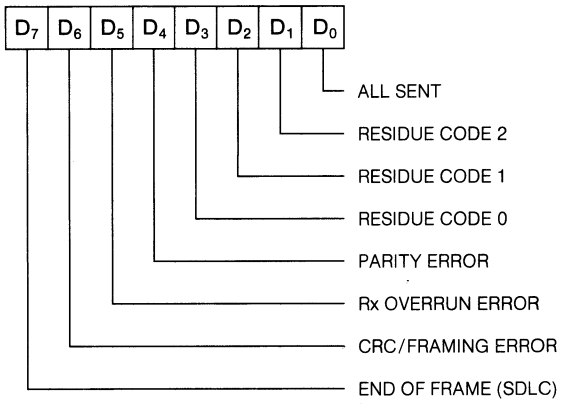
This bit is set when the baud rate generator count is at zero. An external/status interrupt is generated if no other interrupts have been generated. If an interrupt is pending, no interrupt is generated until the interrupt service is completed. If the zero count does not persist until the end of the interrupt under service routine, no interrupt is generated.

D0 **Receive Character Available**

This bit is set when a character is available in the receive buffer. The receiver ports are quadruply buffered (three storage registers and an input shift register) to allow additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. This bit is reset automatically when the receive buffers are all empty. Error status reflects the current byte in the receive buffer. When the receive character bit is set, RR1 should be read and stored to avoid losing any error information. The data received can then be read. Error information does not need to be read with vectored interrupts, since a modified vector is generated if there is an error. Since the receiver port has three storage registers, three characters can be contained in the SCC without a read. A fourth character received without a read causes an overrun error in the last receiver storage register.

Read Register 1

RR1 contains the Special Receive condition status, residue codes, and error conditions.



D7 **End of Frame (SDLC)**

In SDLC mode, this bit indicates reception of a valid ending flag and validation of CRC error and residue codes.

D6 **CRC/Framing Error**

If a framing error occurs (in asynchronous modes), this bit is set (and not latched) only for the character on which it occurred. Detection of a framing error adds in additional half-bit time to the character time so that the framing error will not also be interpreted as a new start bit. In synchronous modes, this bit indicates the result of comparing the CRC to the appropriate value.

D5 **Receiver Overrun Error**

This indicates that more than three characters have been received without a read from the CPU. Only the character that has been written over is flagged with this error, but when this character is read, the error condition is latched until reset by the error reset command (command 6, WR0). If the Status Affects Vector bit is enabled (D0, WR9), the character that has been overrun interrupts with the special receive condition vector.

D4 **Parity Error**

When parity is enabled (D0, WR9), this bit is set for those characters whose parity does not match the programmed one (D1, WR4). The bit is latched so that once an error occurs, the bit remains set until the Error Reset command (command 6, WR0) is given.

D3-D1 **Residue Codes**

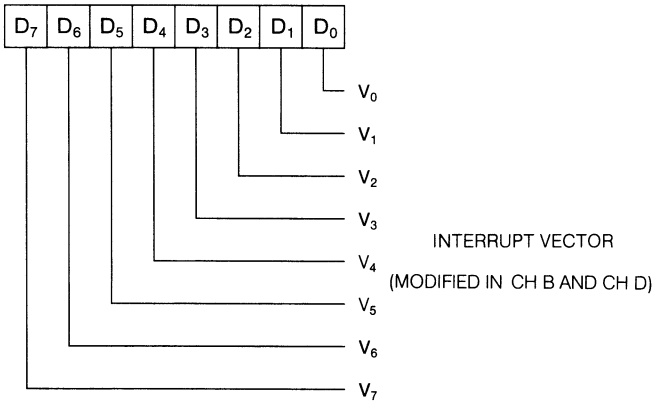
In the SDLC mode, these bits indicate the length of the information field, when the I field is not an integral multiple of the character length. These codes are meaningful upon transfer of the End of Frame bit. If the last character boundary coincides with the boundary of the I field and CRC field, the residue code is always 011.

D0 **All Sent**

This bit is set when all characters have completely cleared the transmitter (XMT Buffer Empty) in asynchronous modes. Interrupts are not caused by transitions of this bit. It is always set in synchronous modes.

Read Register 2 (Channels B and D only)

When read from channel A or C, this register contains the interrupt vector as written into WR2. When read from channel B or D, RR2 contains the interrupt vector modified by the status. When the Status Affects Vector bit is set, status information is programmed into the vector to provide different interrupt handling routines for various SCC conditions (see Figure 3-7, this page, and Figure 3-11, page 3-23). The vector addresses can represent a restart instruction or refer to a table in memory that contains the actual starting address location of the interrupt service routine. The vector address from the SCC contains the low order byte of the interrupt service routine table, with the high order address byte taken from the interrupt register. If no interrupts are pending, V6=1, V5=1, V4=0, or V3=0, V2=1, V1=1 and other bits are as programmed.

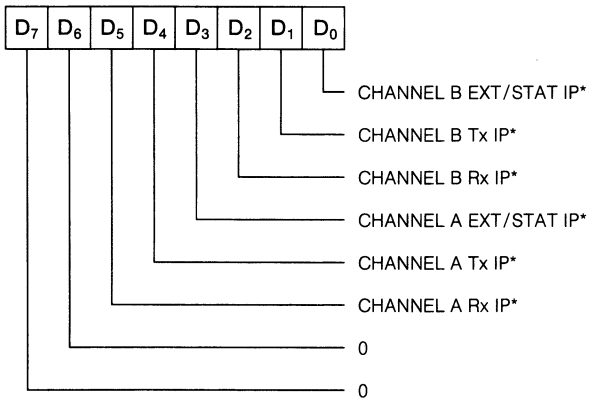


V_3	V_2	V_1	Status High/Status Low = 0
V_4	V_5	V_6	Status High/Status Low = 1
0	0	0	Ch B and D transmit buffer Empty
0	0	1	Ch B and D External/Status Change
0	1	0	Ch B and D Receive Character Available
0	1	1	Ch B and D Special Receive Condition
1	0	0	Ch A and C transmit buffer Empty
1	0	1	Ch A and C External/Status Change
1	1	0	Ch A and C Receive Character Available
1	1	1	Ch A and C Special Receive Condition

Figure 3-7. Status Vector Mode Table

Read Register 3

RR3 contains the status of each interrupt pending bit (channels A and C only). If this register is read from channels B or D, all zeros are returned.



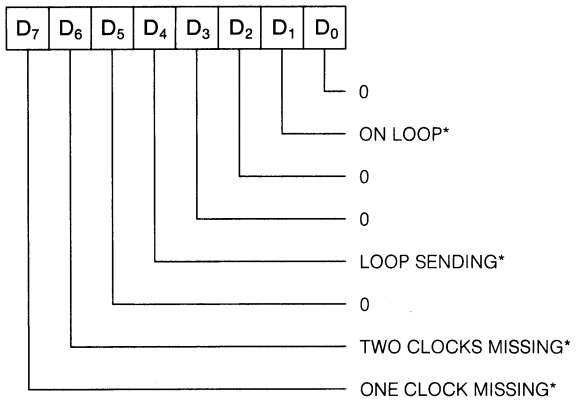
*ALWAYS 0 IN CHANNEL B AND CHANNEL D

Read Register 8

RR8 is the receive buffer.

Read Register 10

RR10 contains miscellaneous transmitter and receiver status parameters. These features are presently not supported by Pro-Log on the 7314A or 7312.



*This feature is not presently supported by Pro-Log.

D7 One Clock Missing

In the FM mode, this bit is set if one clock is missed. In NRZI mode and if the DPLL is disabled, this bit is always reset.

D6 Two Clocks Missing

In the FM mode, this bit is set when the DPLL misses two consecutive clock edges. In the NRZI mode, this bit is always reset.

D4 Loop Sending

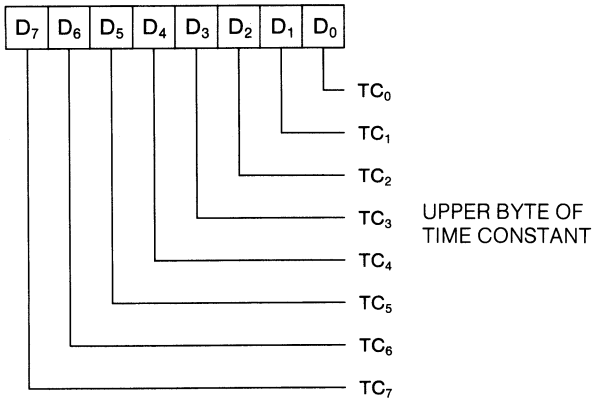
In SDLC loop mode, this bit is set while the transmitter is transmitting on the loop.

D1 On Loop

In SDLC loop mode, this bit is set while the respective SCC is on the loop.

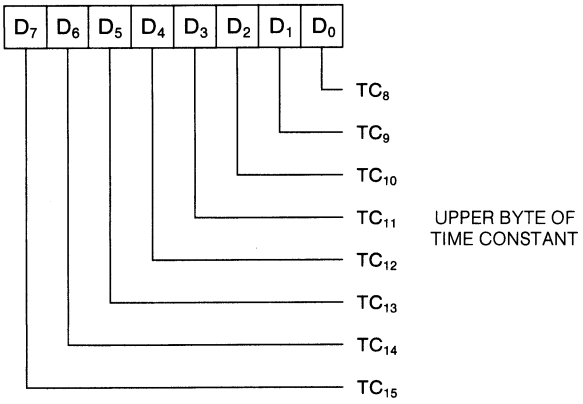
Read Register 12

RR12 contains the lower byte of the baud rate generator time constant as stored in WR12.



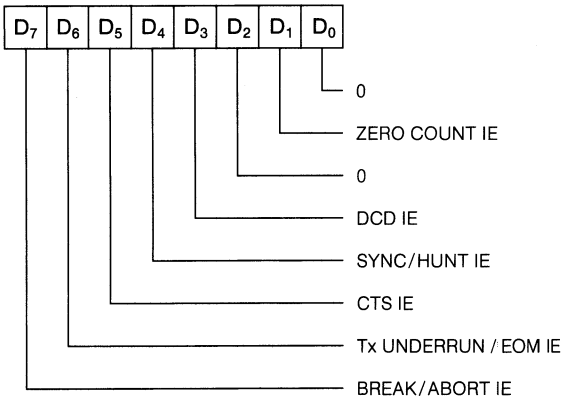
Read Register 13

RR13 contains the upper byte of the baud rate generator time constant as stored in WR13.



Read Register 15

RR15 contains external/status interrupt control information as stored in WR15.



SCC OPERATION

Initialization

The SCC initialization has three stages. In the first stage, the operation modes (i.e., bits per character, parity, etc.) are programmed and the constants (interrupt vector, time constants, etc.) are loaded. In the second stage, the hardware functions are enabled (transmitter, receiver, baud rate generator, etc.). Operating modes should be initialized before hardware functions are enabled. In the third stage, interrupts are enabled. Figure 3-8 shows the order in which SCC registers should be initialized.

REGISTER	DATA	COMMENTS
Stage 1. Modes and Constants		
WR9	XX000000	Channel Reset
WR4	XXXXXXXX	Transmit/Receive Control selects Async or Sync Mode
WR2	XXXXXXXX	Program interrupt vector (optional)
WR3	XXXXXXX0	Selects receiver control. Bit D0 (Rx enable) must be set to 0 at this time
WR5	XXXX0XXX	Selects transmit control. Bit D3 (Tx enable) must be set to 0 at this time.
WR6	XXXXXXXX	Program sync characters
WR7	XXXXXXXX	Program sync characters
WR9	000X0X0X	Select interrupt control Bit D3 (Master interrupt enable) must be set to 0.
WR10	XXXXXXXX	Miscellaneous control (optional)
WR11	0XXXX1XX	Clock control
WR12	XXXXXXXX	Time constant lower byte
WR13	XXXXXXXX	Time constant upper byte
WR14	XXXXXXX0	Miscellaneous control. Bit D0 (BR Generator enable) must be set to 0 at this time.
WR14	XXXSSSS0	This register may require multiple writes if more than one command is used.
Stage 2. Enables		
WR3	SSSSSSS1	Set D0 (Rx enable)
WR5	SSSS1SSS	Set D3 (Tx enable)
WR0	10000000	Reset TxCRC generator
WR14	000SSSS1	BR Generator enable. Set Bit D0 (BR Generator enable)
Stage 3. Interrupt Enables (optional)		
WR15	XXXXX0X0	Enable external interrupts
WR0	00010000	Reset EXT/STATUS twice
WR0	00010000	Reset EXT/STATUS twice
WR1	000XX0XX	Enable receive, transmit, and external interrupts
WR9	000S1S0S	Enable Master Interrupt Bit D3

1 (Set to one)
0 (Set to zero)
X (User's choice)
S (Same as previously programmed)

Figure 3-8. SCC Initialization Order

BAUD RATES

The individual baud rates for each channel are selected by programming the WR12 and WR13 time constants for their respective channels. The formula for determining the appropriate time constant for given baud rate is shown below.

Time constant =

$$\text{(x1 clock):} \left[\frac{4.9162 \times 10^6}{2 \times (\text{Baud Rate})} \right] - 2$$

$$\text{(x16 clock):} \left[\frac{4.9162 \times 10^6}{32 \times (\text{Baud Rate})} \right] - 2$$

Figure 3-9 shows time constants to achieve common baud rates.

	SYNCHRONOUS (x1 Clock)			ASYNCHRONOUS (x16 Clock)		
Baud Rate	CONSTANT Decimal	CONSTANT Hex	% Error	CONSTANT Decimal	CONSTANT Hex	% Error
76.8K*	30	001E	-0-	0	0000	-0-
38.4K*	62	003E	-0-	2	0002	-0-
19.2K	126	007E	-0-	6	0006	-0-
9600	254	00FE	-0-	14	000E	-0-
4800	510	01FE	-0-	30	001E	-0-
2400	1022	03FE	-0-	64	0040	-0-
1800	1361	0551	0.17	83	0053	0.39
1200	2046	07FE	-0-	126	007E	-0-
600	4094	0FFE	-0-	254	00FE	-0-
300	8190	1FFE	-0-	510	01FE	-0-
200	12286	2FFE	-0-	766	02FE	-0-
150	16382	3FFF	-0-	1022	03FE	-0-
134.5	18270	475E	.0007	1140	0474	.0007
110	22340	5744	.0008	1394	0572	.026
75	32766	7FFE	-0-	2046	07FE	-0-
50	49150	BFFE	-0-	3070	0BFE	-0-

*The maximum data rate according to the RS-232-C specification is 20K baud.

Figure 3-9. Time Constants for Common Baud Rates

If four channels are operating simultaneously (7314A), the maximum baud rate possible for each channel is software dependent. Figure 3-10 shows the maximum number of T-states allowable to operate four and eight channels simultaneously asynchronously at 19.2K baud.

		4 Channels	8 Channels	
7314A-1	8088 (5.0 MHz)	650	325	half duplex
		325	162	full duplex
	8085 (6.125 MHz)	406	203	half duplex
		203	101	full duplex
7314A-2	Z80 (2.5 MHz)	325	163	half duplex
		163	81	full duplex
	Z80 (4.0 MHz)	520	260	half duplex
		260	130	full duplex

Figure 3-10. Maximum T-States for Simultaneous Operation

INTERRUPTS

Each SCC has an internal interrupt priority chain. The priorities within the SCC and their relationships on the 7314A and 7312 cards are shown in Figure 3-11. The most common way to operate the SCC is to use the Status Affects Vector feature and program the CPU (Z80 or 8088) for vectored interrupts.

Highest Priority	Description	VIS=0	Vector						VIS=1		
		V7-V4	V3	V2	V1	V0	V7	V6	V5	V4	V3-V0
1	ChA RCV character or	XXXX	1	1	0	X	X	0	1	1	XXXX
	special RCV condition	XXXX	1	1	1	X	X	1	1	1	XXXX
2	ChA XMT Buffer Empty	XXXX	1	0	0	X	X	0	0	1	XXXX
3	ChA External/Status	XXXX	1	0	1	X	X	1	0	1	XXXX
4	ChB RCV character or	XXXX	0	1	0	X	X	0	1	0	XXXX
	special RCV condition	XXXX	0	1	1	X	X	1	1	0	XXXX
5	ChB XMT Buffer Empty	XXXX	0	0	0	X	X	0	0	0	XXXX
6	ChB External/Status	XXXX	0	0	1	X	X	1	0	0	XXXX
7	ChC RCV character or	XXXX	1	1	0	X	X	0	1	1	XXXX
	special RCV condition	XXXX	1	1	1	X	X	1	1	1	XXXX
8	ChC XMT Buffer Empty	XXXX	1	0	0	X	X	0	0	1	XXXX
9	ChC External/Status	XXXX	1	0	1	X	X	1	0	1	XXXX
10	ChD RCV character or	XXXX	0	1	0	X	X	0	1	0	XXXX
	special RCV condition	XXXX	0	1	1	X	X	1	1	0	XXXX
11	ChD XMT Buffer Empty	XXXX	0	0	0	X	X	0	0	0	XXXX
12	ChD External/Status	XXXX	0	0	1	X	X	1	0	0	XXXX

Figure 3-11. Vectored Interrupt Priority

The 7314A and 7312 use the STD BUS interrupt priority chain. Priority is based on position in the card rack with the highest priority on the right. (Note that some manufacturers make card racks with the highest priority on the left.) The 7314A, 7312, and other cards in a system that make use of the priority chain should be installed in the card rack according to their priority.

The 7314A and 7312 have a fixed priority for the various interrupt sources on the card, but as a card it may be higher or lower than other cards.

Cards that use the priority chain must be placed next to each other with no unused card slots in between. Other cards that do not use the priority chain can be installed between cards using the priority chain but these other cards must pass on the priority chain: they must connect pins 51 and 52 (priority chain out and priority chain in) together onboard. This is standard practice on Pro-Log STD BUS cards.

The cards that use the priority chain, as a group, can be placed anywhere in the card rack and other cards can be placed anywhere on either side of the group. Other cards can have empty slots between them or can be placed in the group provided they pass on the priority chain.

The number of devices that can participate in the PCI-PCO interrupt serial daisy chain is system dependent. STD BUS specifications allow the following settling times (refer to Interrupt Acknowledge timing diagrams):

- **8088 CPU operation: (7314A-1, 7312-1)**

  MCSYNC* (1) low to INTAK* (2) low: tS = 800 ns minimum.

- **Z80 CPU operation: (7314A-2, 7312-2)**

  STATUS1* low to INTAK* low: 2.5 MHz: tS = 850 ns minimum; 4.0 MHz: tS = 500 ns minimum

- **8085 CPU operation: (7314A-2, 7312-2)**

  IORQ*, S1*, S0* low to INTAK* low: tS = 270 ns minimum

The actual settling times provided are system dependent. The following formula can be used as a guideline to determine the number of devices that can participate in the interrupt daisy chain.

$$tS = t \begin{bmatrix} \text{INTAK* 1 of the 8088} \\ \text{or INTAK* of the Z80} \\ \text{or IORQ* • S1* • S0* of the 8085} \end{bmatrix} \text{to PCO, highest priority device}$$

+ Σt (PCI to PCO for each device)

+ Σt (PCI to INTAK* setup time, lowest priority device)

Sample calculation for the number of 7314A-(n)'s operating in a typical 8088 system
(where Σt (PCI to PCO for each device)
= n (PCI to PCO for one 7314A)):

800 ns = 275 ns + n(52 ns) + 155 ns
n = 7 7314A-2 cards

For the 7314A-2 operating in typical Z80 systems:
n = 32 cards (2.5 mHz); n = 18 cards (4.0 MHz)

For the 7314A-2 operating in a typical 8085A system:
n = 1 card

Cards that do not have look-ahead capability may have a longer delay from PCI to PCO and the number of devices on the chain may be greatly reduced.

Interrupt operation options for the 7314A and 7312 are available for systems that do not use vectored interrupts.

1. Interrupts are not used at all in the system:

   The interrupt pending bits of RR3 are monitored. When one of the bits is set (RR3 is not equal to 0), the vector is read from RR2 (having previously initialized the Vector Includes Status mode). The vector is used to determine the appropriate response. The interrupt acknowledge disable jumper (W2 9-10) should be installed (with W2 7-8 removed) for this mode of operation.

2. Other cards in the system generate interrupts but only provide 8080-type restart vectors:

   Program the SCC WR2 with a restart instruction as the vector. The Vector Includes Status mode is not used. W2 7-8 should be installed (W2 9-10 removed). When an interrupt is generated by the 7314A or 7312, a restart instruction is provided.

3. Other cards in the system generate interrupts but do not provide vectors:

   Two options are available.

   a. Disable the 7314A or 7312 vector response by removing W2 7-8, installing W2 9-10, and using the SCCs in polled mode only. RR3 is read to determine that an interrupt is pending and RR0 of the interrupting channel is read to determine the specific cause of interrupt.

   b. The 7314A and 7312 have interrupt request input connectors that are used in conjunction with the SCC to generate unique vectors for system cards that do not provide vectors. The interrupt request signals of up to four other cards with a 7314A and two cards for the 7312 are connected to the SCC sync lines. An external interrupt request generates an external/ status interrupt on the selected SCC channel and the corresponding vector (the Status Affects Vector mode) is provided during an interrupt acknowledge cycle. For this method of operation, the 7314A or 7312 interrupt acknowledge response should be enabled (W2 7-8 installed, W2 9-10 removed). This method can only be used when the respective SCC channel is operating in asynchronous mode.

4. PCI-PCO are not used for interrupt priority.

   If another scheme of interrupt prioritization is used (i.e., a Pro-Log 7320 Priority Interrupt Controller Card or a system with master-slave 8259 PICs), the 7314A and 7312 INTRQ* to the STD BUS can be disconnected by removing W2 11-12. On the 7314A, the two SCC interrupts can appear separately at J1 (W2 13-14 removed) or they can be wire-ORed (W2 13-14 installed).

RS-232-C SIGNALS

The EIA RS-232-C standard defines connector pin functions and signal levels for serial data communications. It is applicable for the interchange of data, timing, and control signals, with a signal ground. These signals are sent between Data Terminal Equipment (DTE) and Data Communication Equipment (DCE). When interfacing two pieces of serial communication equipment, one must be configured as DTE and the other as DCE. The DTE is considered the main transmitter and the DCE the main receiver; thus the RS-232-C signals are defined in terms of the DTE.

The following are short definitions of each signal provided on the 7314A and 7312 RS-232-C interfaces. JX-X refers to the pin on the RS-232-C connectors J2, 3, 4, and 5.

TD*, JX-3: Transmitted Data signals are sent from the DTE to the DCE.

RD*, JX-5: Received Data signals come from the DCE and are sent to the DTE.

RTS, JX-7: Request To Send is a handshake line sent from the DTE to the DCE. The DTE is telling the DCE it is ready to transmit.

CTS, JX-9: Clear To Send is the handshake line sent from the DCE to the DTE. When the DCE is ready to receive, it notifies the DTE through this signal.

DSR, JX-11: Data Set Ready is a control line sent from the DTE to the DCE. When the line is active low, the DTE is on-line, in data transmission mode, and data can be sent to it. When DTR is high, the DTE is off-line in voice or local mode, and data should not be sent to it.

DTR, JX-14: Data Terminal Ready is a control line sent from the DTE to the DCE. When the line is active low, the DTE is on-line and data can be sent to it from the DCE. When DTR is high, the DTE is off-line (local mode) and data should not be sent to it.

Figure 3-12 shows an example of signal flow from DTE equipment (on the right) to the 7314A or 7312 card in the DCE configuration (on the left). Note that DTE transmit data (TD*) is the DCE receive data (RD*); DTE request to sent (RTS) is the DCE clear to send (CTS); DTE data terminal ready (DTR) is the DCE data set ready (DSR); and so on. Figure 3-13 shows an example of signal flow from DCE equipment (on the right) to the 7314A or 7312 card in the DTE configuration on the left.

Some terminals and printers may require an alternate handshaking scheme as follows:

When the DTE is ready to send data, it pulls its RTS line active. This line is connected to the DSR line of the DCE. The DCE responds by pulling its DTR line active, which is connected to the CTS line of the DTE. When the DCE is ready to send data, it pulls its RTS line active. This line is connected to the DSR line of the DTE. The DTE responds by pulling its DTR line active, which is connected to the DTR line of the DCE. In this manner, the DTE verifies that the DCE is listening before sending data and the DCE verifies that the DTE is listening before sending data. This configuration can be achieved by wiring the jumpers as illustrated by dotted lines in Figure 3-12.

The CTS* and DCD* pins automatically enable/disable the transmitter and receiver if auto-handshaking is selected (WR3 D5 is set). If the transmitter is disabled while transmitting a character, the transmission of the character is completed.

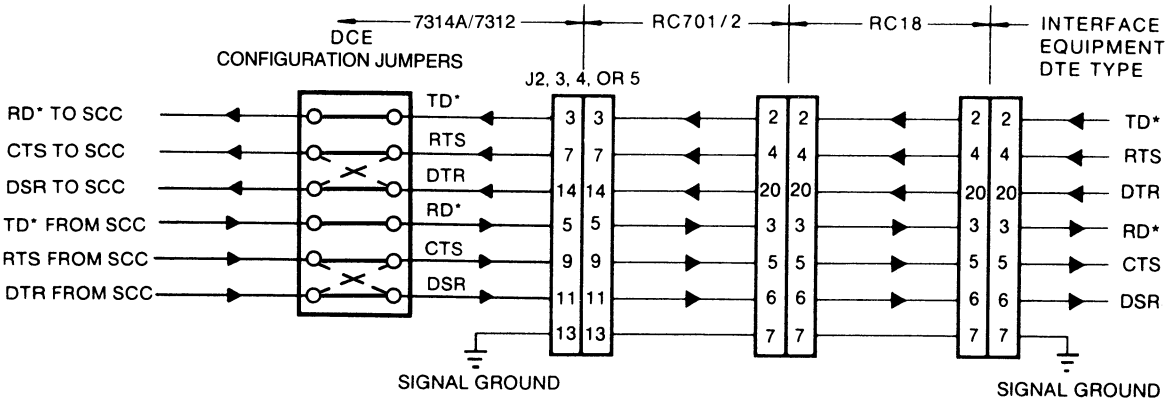


Figure 3-12. Signal Flow in DCE Configuration

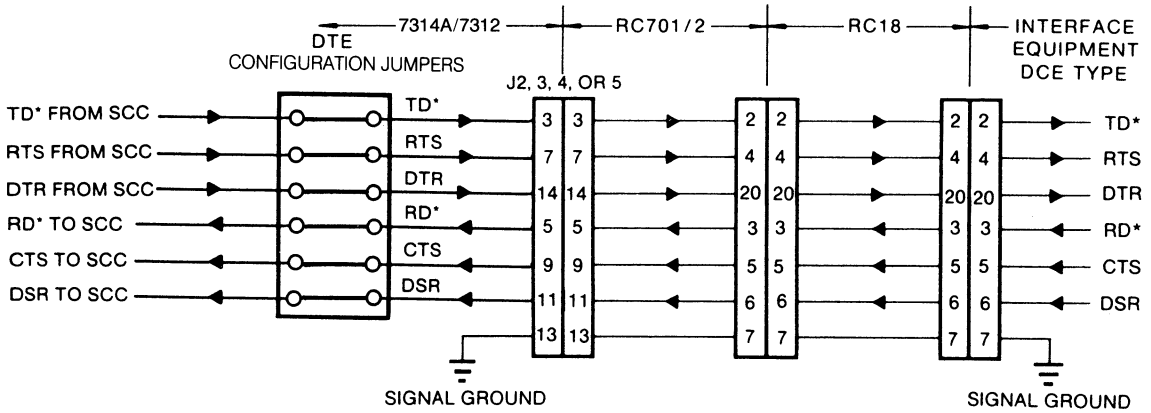


Figure 3-13. Signal Flow in DTE Configuration

Three elements control handshaking for the RS-232-C port: the SCC, the processor, and jumpers W3-6. The jumpers select to which RS-232-C line each handshake pin of the SCC is connected. The SCC can control some of the handshake lines and the processor can control all or some of them through the SCC registers, depending upon whether or not auto-handshaking is enabled. Figure 3-14 shows which lines can be controlled by the SCC.

The CTS* and DCD* pins automatically enable/disable the transmitter and receiver if auto-handshaking is selected (WR3 D5 is set). If the transmitter is disabled while transmitting a character, the transmission of the character is completed.

If handshake interrupts are enabled and auto-handshaking is not enabled, the CTS* and DCD* pins must be monitored by the processor or they can generate interrupts. CTS* and DCD* are monitored through RR0. The DTR* and RTS* pins are controlled by the processor through WR5.

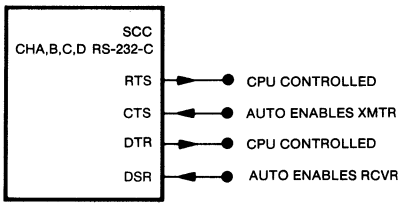


Figure 3-14. SCC RS-232-C Handshake Capability

Section 4
OPERATING SOFTWARE

INTRODUCTION

This section contains hardware-level subroutine modules with which to operate the 7314A and 7312 cards. The software in this section can be used without license from Pro-Log. Although tested and believed correct, this software is not represented to be free from errors or copyright infringement, or appropriate for any specific application.

The subroutines are presented in two forms. The Z80 version is in STD mnemonics and was developed using Pro-Log's CASE system. The 8088 version is in Intel mnemonics generated on an IBM PC using the IBM PC Macroassembler. A copy of the source listing file is included at the end of this section, following the flowcharts. See Pro-Log's Application Note #156, provided with this manual, for more information about serial communications formats.

MEMORY ADDRESSES

Full memory addresses are given. If your system cannot use the memory addresses in the 7314A and 7312 software package, simply change the memory page addresses, as required, when loading these modules into your system.

I/O PORT ADDRESSES

The 7314A and 7312 I/O ports are assigned preferred hexadecimal addresses 0B0-0B7 (B0-B7 for the Z80) for compatibility with other Series 7000 cards. Section 2 shows how to remap these addresses if necessary. This software can be used by changing the port addresses when loading the program modules into your system.

DESCRIPTION

The software provided in this manual is written to display the various functions of the SCC. Some jumpers may need to be moved depending upon which serial communication format is used. This software demonstrates programming the SCC to transmit and receive data both asynchronously for polled operation on channels A and B, and in vectored interrupt mode using SDLC format on channels C and D. The latter can be modified to operate with channels A and B for the 7312 by changing the port addresses.

There are 13 subroutines, most of which are subroutines within subroutines and not accessed directly. Only seven of them need to be accessed directly to perform serial communications. Others are useful as building blocks in the user's own software for serial communication. All of these are shown in Figure 4-1.

This software is not complete in itself. There is no software provided to process errors in data reception or changes in incoming handshake lines, since these are largely application dependent. If used as a package with user-supplied software, these subroutines can greatly simplify the use of the serial ports. Two examples that use these subroutines are presented at the end of the program. The (ASYNC COMM) routine transfers a block of data asynchronously from channel A to channel B. When the transmission is finished, the program loops at END ASYNC. The (SDLC COMM) routine transfers a block of data in the SDLC format and loops at END SDLC when finished. Both routines require that the byte count and first character address mailboxes be initialized in RAM. The (SDLC COMM) routine also requires the vector table to be entered in RAM.

NAME	DESCRIPTION	MAIN SUBROUTINE	MAY BE USED BY ITSELF	NOT USEFUL BY ITSELF
(INIT ASYN COMM)	Initialize asynchronous communications interface	X		
(INIT SCC X)	Initialize specified SCC channel			X
(INIT SCC CH)	Initialize one channel of SCC		X	
(XMT CH A ASYNC)	Transmit a character asynchronously (Ch. A)	X		
(RCV CH B ASYNC)	Receive a character asynchronously (Ch. B)	X		
(INIT SDLC COMM)	Initialize SDLC communications interface	X		
(INIT INTER)	Initialize interrupt		X	
(XMT CH C SDLC)	Transmit first character in SDLC format (Ch. C)	X		
(XMT BLK C)	Transmit a character in SDLC format	X		
(RCV CH D SDLC)	Receive a character (Ch. D)		X	
(SPCL RCV D SDLC)	Determine error/EOF (Ch. D)		X	

Figure 4-1. Subroutine Types

If the software provided is used as a package, only a few requirements must be met by the user. The seven main subroutines and their requirements are given in Figure 4-2. The following list details the requirements and the points to be considered before implementing the software.

- The user may wish to provide software for handshake control or for responding to changes in incoming handshaking.
- Processing of errors in received data requires user-supplied software.
- In received data routines, the character is simply received and stored. Jump to Subroutine instructions are provided to jump to the user-supplied subroutines. The user's subroutine may then respond to the character according to the requirements of the application.
- The baud rates and various other factors selected by the initialization routine should be checked to see that they match the application. The initialized values are given in the description of the (INIT COMM) routines.

NAME	DESCRIPTION	USE REQUIREMENTS
(INIT ASYN COMM)	Initialize SCC channels A and B	None
(XMT CH A ASYNC)	Transmit a character asynchronously (Ch. A)	Data in memory
(RCV CH B ASYNC)	Receive and store a character or determine transmission error (Ch. B)	User provides software to process received data and errors
(INIT SDLC COMM)	Initialize SCC channels C and D	None
(XMT CH C SDLC)	Transmit a character in SDLC format (Ch. C)	Control and data in memory
(RCV CH D SDLC)	Receive and store a character (Ch. D)	
(SPCL RCV D SDLC)	Determine SDLC transmission error or end of frame (Ch. D)	User provides software to process errors and data frame

Figure 4-2. Main Subroutines Use Requirements

SUBROUTINE SPECIFICATIONS

(INIT ASYN COMM)

ENTRY ADDRESS: 0100
ENTRY : None
EXIT : SCC channels A and B initialized
ALTERS : None
CALLS : (INIT SCC A), (INIT SCC B)
STACK : 6

FUNCTIONAL DESCRIPTION:

This subroutine initializes channels A and B for polled asynchronous communications. The initialized conditions should be checked to be sure they match the application. The initial baud rate for channels A and B is 4800 and the auto-handshake enables mode is used.

(INIT SCC X)

ENTRY ADDRESS: 0100 (channel A)
0128 (channel B)
0210 (channel C)
022A (channel D)
ENTRY : None
EXIT : Channel X initialized
ALTERS : A, B, C, H, L
CALLS : (INIT SCC CH)
STACK : None

FUNCTIONAL DESCRIPTION:

This subroutine initializes one channel of an onboard SCC. (INIT SCC A) and (INIT SCC B) initialize for asynchronous communications, and (INIT SCC C) and (INIT SCC D) initialize channels C and D for SDLC communications.

(INIT SCC CH)

ENTRY ADDRESS : 0140
ENTRY : SCC control port address in register C, channel initialization starting address in HL.
EXIT : SCC channel initialized
ALTERS : A, B, H, L
CALLS : Requires initialization data table
STACK : None

FUNCTIONAL DESCRIPTION:

This subroutine initializes one channel of one SCC. The channel is selected by loading register pair HL with the appropriate address (see Entry Requirements). If vectored interrupts are used, write register 2B must contain the interrupt vector for both SCC channels A and B, and write register 2D must contain the interrupt vector for SCC channels C and D.

(XMT CH A ASYNC)

ENTRY ADDRESS : 0180
ENTRY : Data byte count in XMT COUNT A mailbox, address of first data to be transmitted in XMT BUF A mailbox, byte count decrement address incremented
EXIT : Byte transmitted
ALTERS : None
CALLS : None
STACK : 8

FUNCTIONAL DESCRIPTION:

This subroutine asynchronously transmits a block of data of up to 255 bytes. The byte count must be stored in the XMT COUNT A mailbox, and the address of the first data to be transmitted must be stored in the XMT BUF A mailbox. This subroutine updates these mailboxes each time a byte is transmitted. Polled mode is used: the program loops until SCC Read Register 0 (RR0) indicates that the channel is ready to transmit.

(RCV CH B ASYNC)

ENTRY ADDRESS : 01C0
ENTRY : Current character store address in RCV BUF B mailbox
EXIT : Only data byte stored, address and character store received, channel B read register 1 (RR1) updated
ALTERS : None
CALLS : (FRAMING ERROR), (RCV OVRN ERROR), (PARITY ERROR), user-supplied subroutine
STACK : 4

FUNCTIONAL DESCRIPTION:

This is a polled routine that receives an asynchronously transmitted character and stores it in memory. This subroutine loops until a character is received. The user must supply a program (PROCESS DATA) to interpret the received data string. The receivers are enabled at initialization. If a framing error, receiver overrun, or parity error occurs, the program exits to a user-supplied subroutine.

(INIT SDLC COMM)

ENTRY ADDRESS : 0200
ENTRY : None
EXIT : SCC channels C and D initialized
ALTERS : None
CALLS : (INIT SCC C), (INIT SCC D), (INIT INTER)
STACK : 6

FUNCTIONAL DESCRIPTION:

This subroutine initializes channels C and D for vector interrupt SDLC communications at 19.2K baud. The initialized conditions should be checked to be sure they match the application.

(INIT INTER)

ENTRY ADDRESS : 02C0
ENTRY : None
EXIT : SCC interrupts enabled
ALTERS : None
CALLS : None
STACK : 2

FUNCTIONAL DESCRIPTION:

This subroutine selects and enables the vectored interrupt mode.

(XMT CH C SDLC)

ENTRY ADDRESS : 02E0
ENTRY : None
EXIT : Transmitted secondary address byte, reset XMT CRC generator, reset overrun/end of message bit
ALTERS : None
CALLS : None
STACK : 2

FUNCTIONAL DESCRIPTION:

This subroutine transmits the first byte of a block of data using the SDLC format. When data can be sent, the transmit CRC generator is reset, the secondary address (see Figure 3-1) is sent, and the reset overrun/end of message latch is reset. Thereafter, whenever the transmitter indicates it is ready, via the interrupt, the program responds by sending another byte from memory using the (XMT BLK C) subroutine.

(XMT BLK C)

ENTRY ADDRESS : 0300
ENTRY : HL pointing to data byte count in XMT COUNT C mailbox, address of first data to be transmitted in XMT BUF C
EXIT : One byte transmitted, byte count decremented, address incremented, lower priority interrupt enabled
ALTERS : None
CALLS : None
STACK : 6

FUNCTIONAL DESCRIPTION:

This is an interrupt service routine that transmits a byte of data using the SDLC format. The data transmitted is located at the address contained in the XMT BUF C mailbox. The address and byte count are updated and lower priority interrupts are enabled. After the last data byte has been sent, the transmit interrupt is disabled.

(RCV CH D SDLC)

ENTRY ADDRESS : 0340
ENTRY : Current character store address in RCV BUF B
EXIT : One received data byte stored and character store address updated
ALTERS : None
CALLS : None
STACK : 2

FUNCTIONAL DESCRIPTION:

This is an interrupt service routine that receives and stores synchronously transmitted SDLC frames (see Figure 3-1). The receivers are enabled at initialization. If errors are detected (SDLC CRC errors or receiver overrun), a Special Receive interrupt is generated. When the final flag is received signaling the end of the frame, a Special Receive interrupt occurs. A subroutine must be supplied to process the frame. Lower priority interrupts are enabled at the end of this routine.

(SPCL RCVD SDLC)

ENTRY ADDRESS : 035B
ENTRY : None
EXIT : Channel D status mailbox updated
ALTERS : None
CALLS : (SDLC CRC ERROR), (RCV OVRN CH D), (END OF FRAME), user-supplied subroutines
STACK : 4

FUNCTIONAL DESCRIPTION:

This is an interrupt service routine that is activated when an SDLC CRC error, receiver overrun, or end of frame is detected. The user must supply subroutines to process the errors and interpret the data received. After processing, the error code is reset and lower priority interrupts are enabled.

PAGE / LINE	01	02	03	04
00 | 0F	(INIT ASYN COMM)	(INIT SDLC COMM)	(XMT BLK C)	
10 | 1F	(INIT SCC A)	(INIT SCC C)		
20 | 2F	(INIT SCC B)			
30 | 3F		(INIT SCC D)	LAST SDLC	
40 | 4F	(INIT SCC CH)	SDLC INIT CH C (DATA TABLE)	(RCV CH D SDLC)	
50 | 5F	ASYNC INIT (DATA TABLE)			
60 | 6F			(SPCL RCVD SDLC)	
70 | 7F				
80 | 8F	(XMT CH A ASYNC)	SDLC INIT CH D (SDLC COMM)	(ASYNC COMM)	
90 | 9F				
A0 | AF	LAST ASYNC		(SDLC COMM)	
B0 | BF				
C0 | CF	(RCV CH B ASYNC)	(INIT INTER)		
D0 | DF				
E0 | EF		(XMT CH C SDLC)		
F0 | FF				

Figure 4-3. 1K Program Memory Map

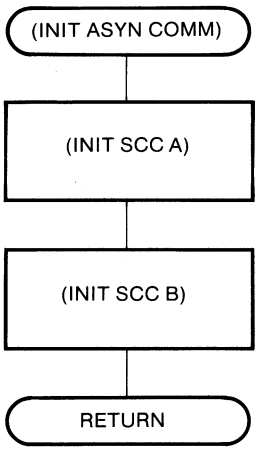


Figure 4-4. (INIT ASYN COMM) Flowchart

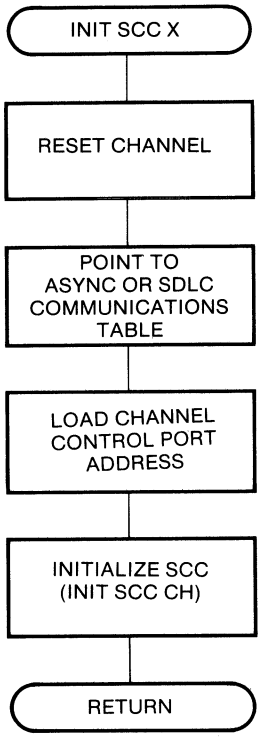


Figure 4-5. (INIT SCC X) Flowchart

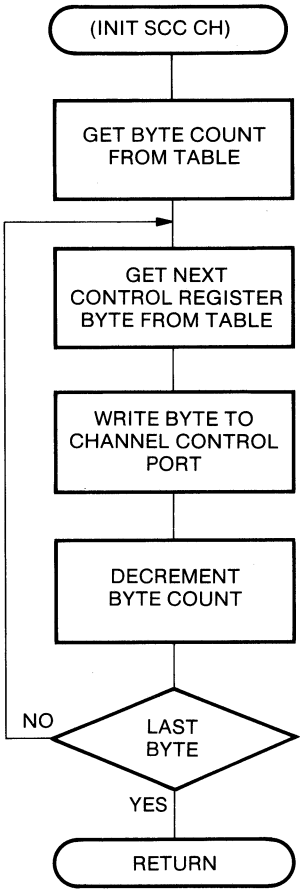


Figure 4-6. (INIT SCC CH) Flowchart

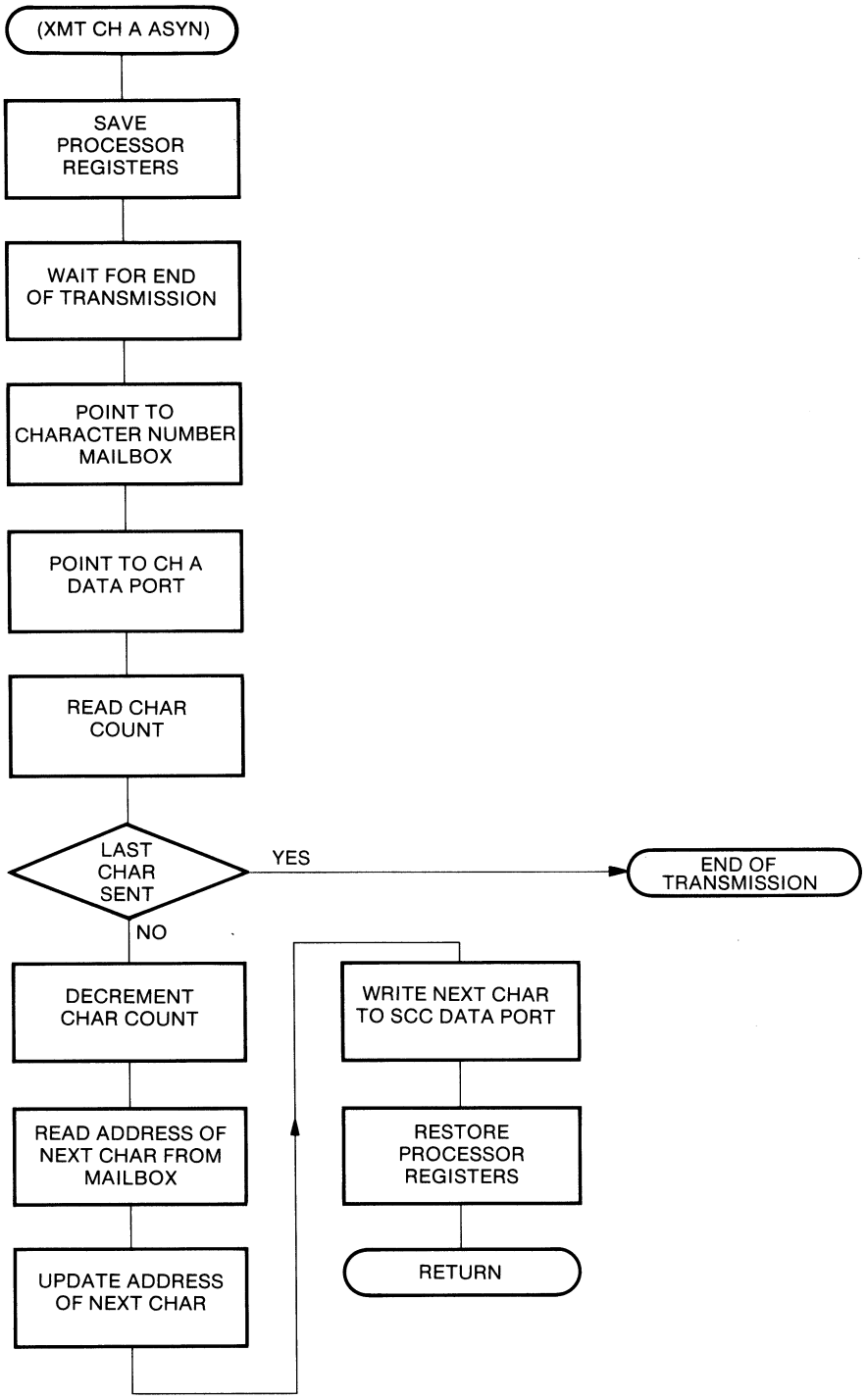


Figure 4-7. (XMT CH A ASYNC) Flowchart

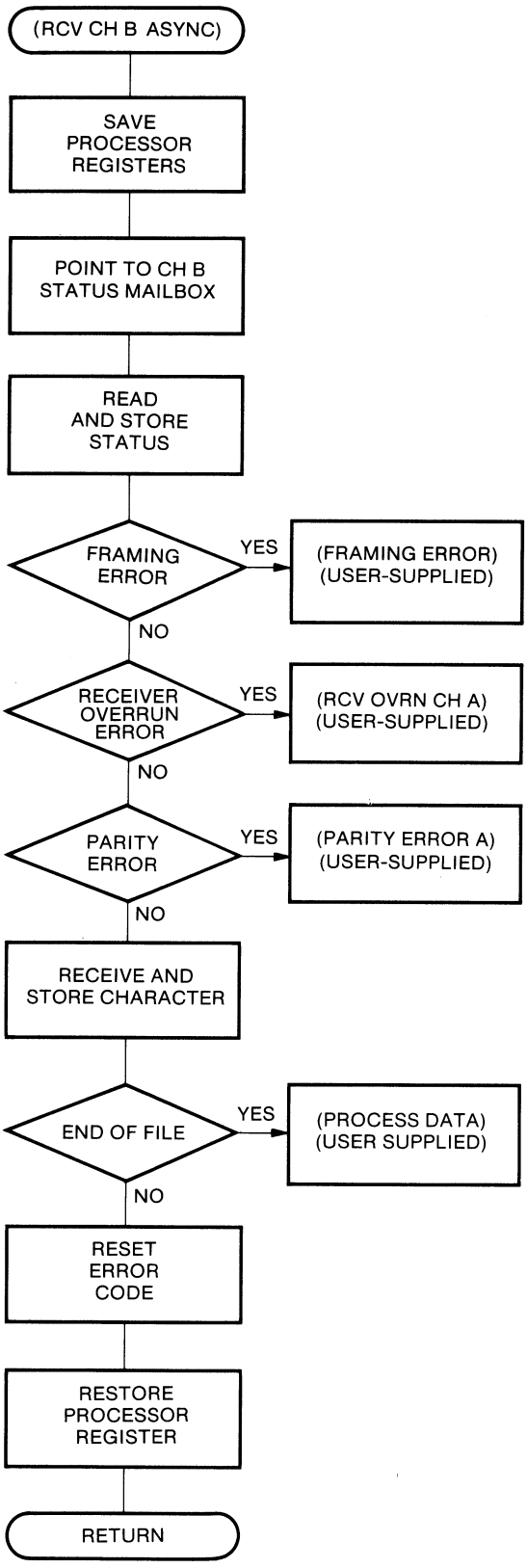


Figure 4-8. (RCV CH B ASYNC) Flowchart

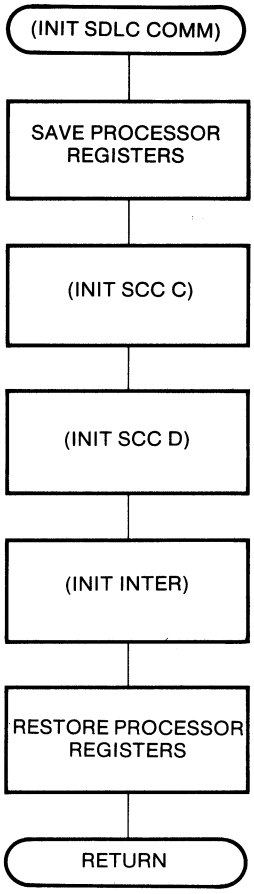


Figure 4-9. (INIT SDLC COMM) Flowchart

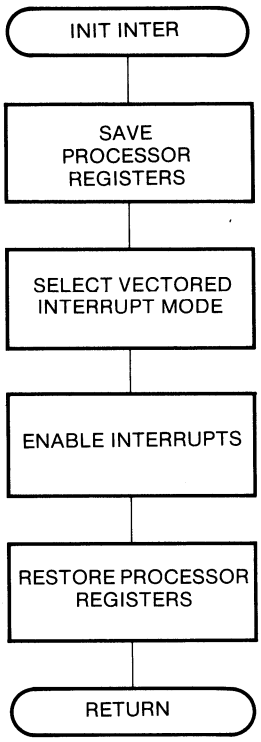


Figure 4-10. (INIT INTER) Flowchart

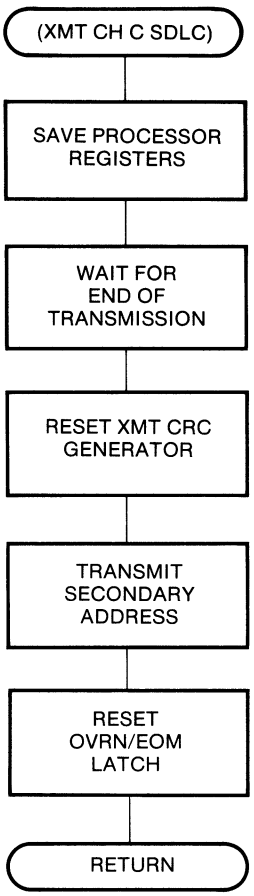


Figure 4-11. (XMT CH C SDLC) Flowchart

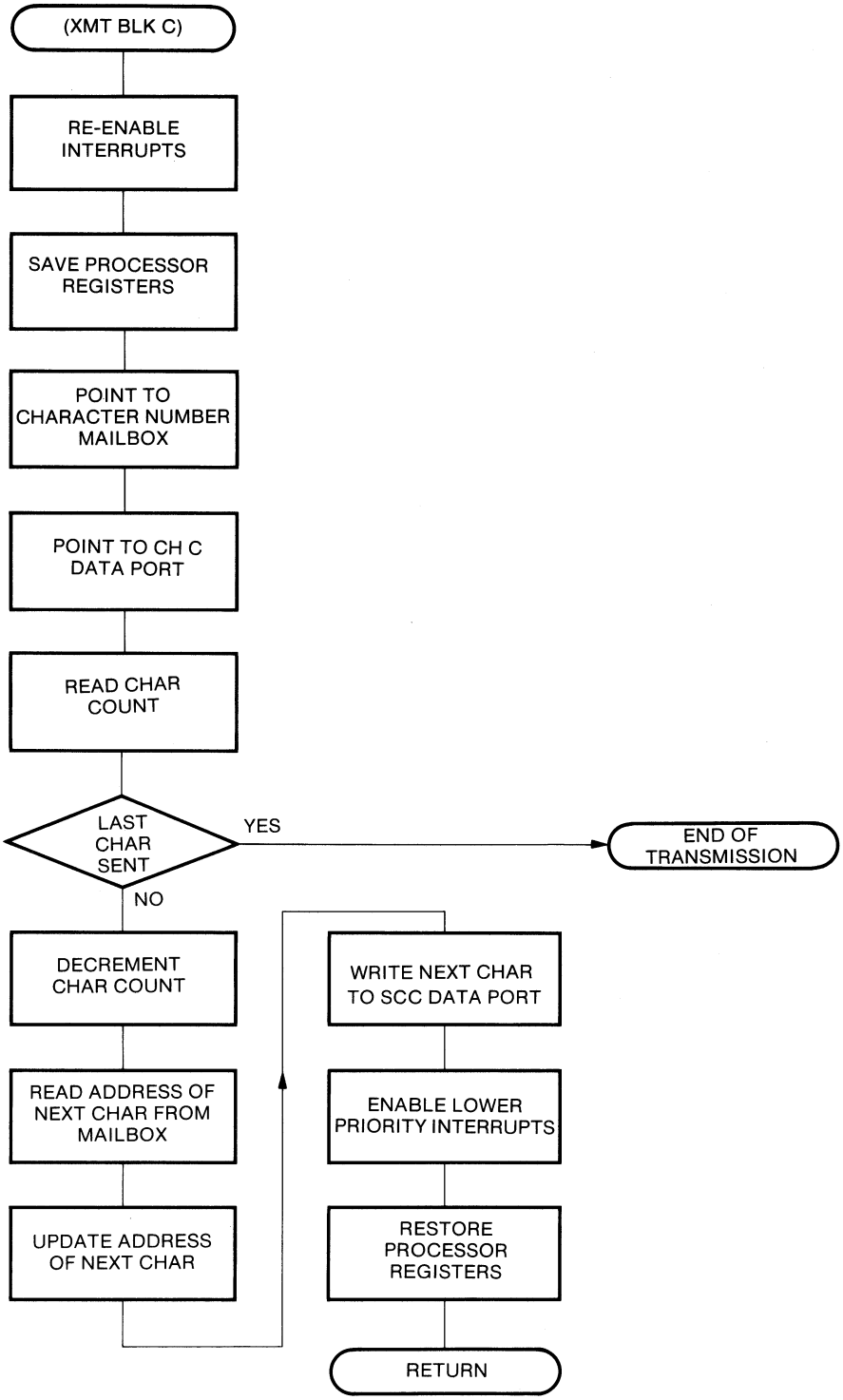


Figure 4-12. (XMT BLK C) Flowchart

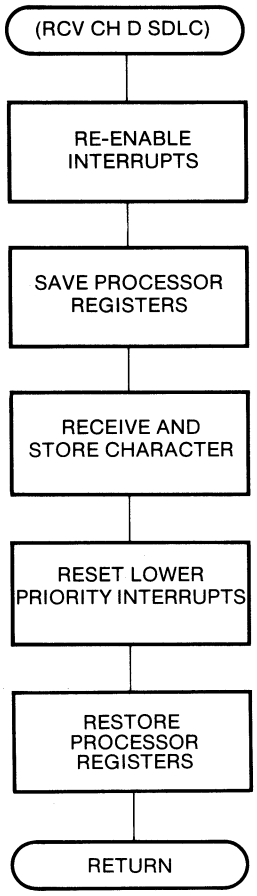


Figure 4-13. (RCV CH D SDLC) Flowchart

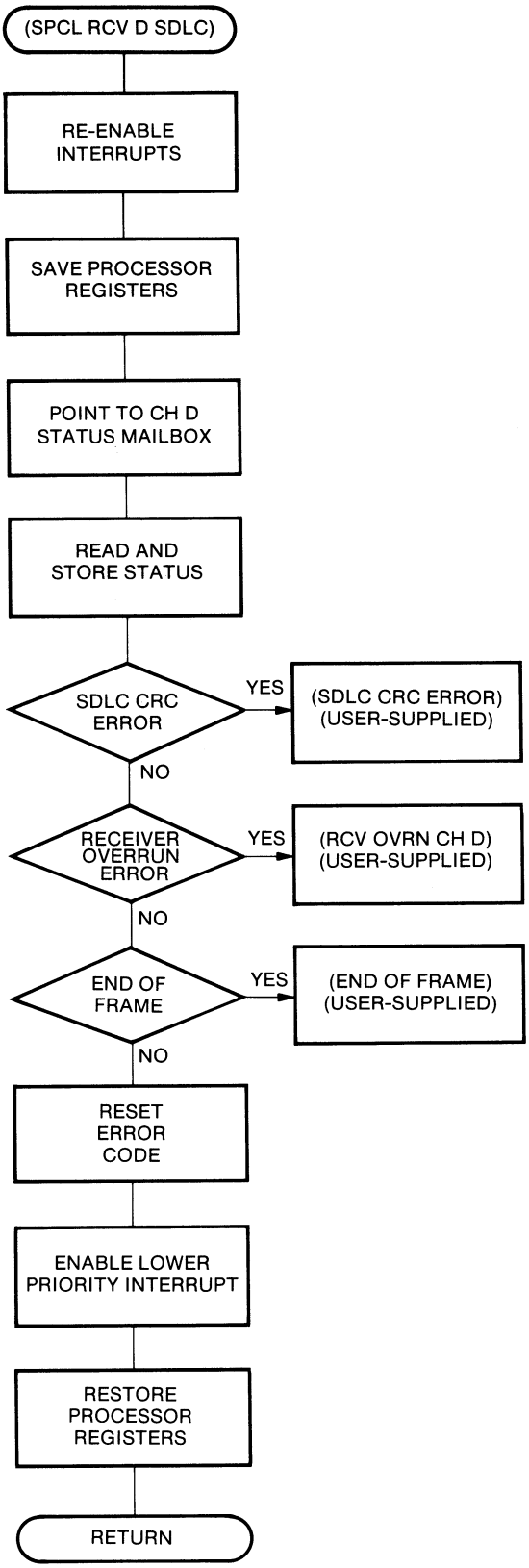


Figure 4-14. (SPCL RCVD SDLC) Flowchart

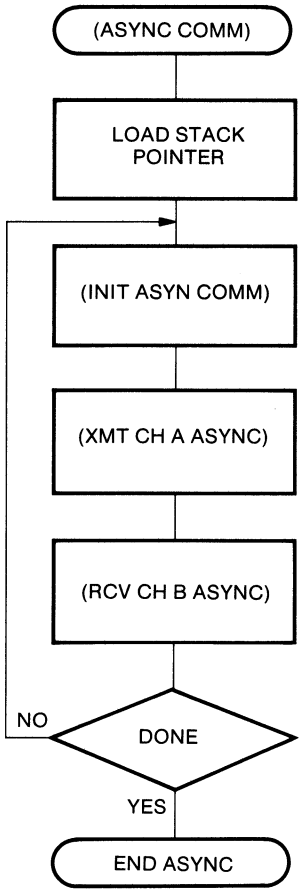


Figure 4-15. (ASYNC COMM) Flowchart

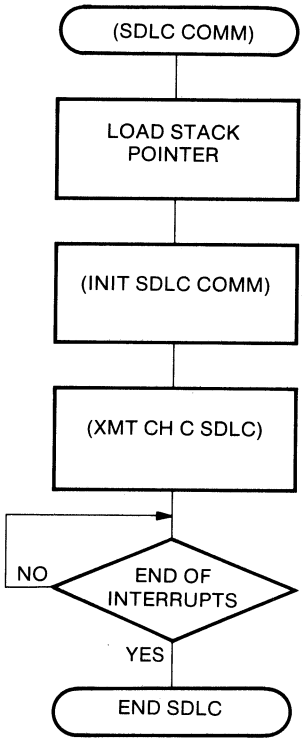


Figure 4-16. (SDLC COMM) Flowchart

```
ADRdata!      LABEL        !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0100 F5!(INIT ASYN COMM)!PSP   !AF                  !Save processor registers
  01 C5!                 !PSP   !BC                  !
  02 E5!                 !PSP   !HL                  !
  03 CD!                 !JS    !UN                  !Initialize SCC channels A
  04 10!                 ! ---- !(INIT SCC A)        !  and B for asynchronous
  05 01!                 ! ---- !                    !  communications
  06 CD!                 !JS    !UN                  !
  07 28!                 ! ---- !(INIT SCC B)        !
  08 01!                 ! ---- !                    !
  09 E1!                 !PLP   !HL                  !Restore processor registers
  0A C1!                 !PLP   !BC                  !
  0B F1!                 !PLP   !AF                  !
  0C C9!    ----------   !RTS   !UN                  !Return
  0D FF!                 !      !                    !
  0E FF!                 !      !                    !
  0F FF!                 !      !                    !
0110 3E!(INIT SCC A)     !LDAI  !                    !Reset channel A
  11 09!                 ! ---- !09 hex              !
  12 D3!                 !OPA   !                    !
  13 B1!                 ! ---- !B1 hex              !
  14 3E!                 !LDAI  !                    !
  15 80!                 ! ---- !80 hex              !
  16 D3!                 !OPA   !                    !
  17 B1!                 ! ---- !B1 hex              !
  18 21!                 !LDPI  !HL                  !Initialize channel A for
  19 50!                 ! ---- !ASYNC INIT          !  asynchronous communications
  1A 01!                 ! ---- !                    !
  1B 0E!                 !LDRI  !C                   !Channel A control port
  1C B1!                 ! ---- !B1 hex              !  address = B1
  1D CD!                 !JS    !UN                  !
  1E 40!                 ! ---- !(INIT SCC CH)       !
  1F 01!                 ! ---- !                    !
0120 C9!    ----------   !RTS   !UN                  !Return
  21 FF!                 !      !                    !
  22 FF!                 !      !                    !
  23 FF!                 !      !                    !
  24 FF!                 !      !                    !
  25 FF!                 !      !                    !
  26 FF!                 !      !                    !
  27 FF!                 !      !                    !
0128 3E!(INIT SCC B)     !LDAI  !                    !Reset channel B
  29 09!                 ! ---- !09 hex              !
  2A D3!                 !OPA   !                    !
  2B B3!                 ! ---- !B3 hex              !
  2C 3E!                 !LDAI  !                    !
  2D 40!                 ! ---- !40 hex              !
  2E D3!                 !OPA   !                    !
  2F B3!                 ! ---- !B3 hex              !
0130 21!                 !LDPI  !HL                  !Initialize channel B for
  31 50!                 ! ---- !ASYNC INIT          !  asynchronous communicatiion
  32 01!                 ! ---- !                    !
  33 0E!                 !LDRI  !C                   !Channel B control port
  34 B3!                 ! ---- !B3 hex              !  address = B3
  35 CD!                 !JS    !UN                  !
  36 40!                 ! ---- !(INIT SCC CH)       !
  37 01!                 ! ---- !                    !
  38 C9!    ----------   !RTS   !UN                  !Return
  39 FF!                 !      !                    !
  3A FF!                 !      !                    !
  3B FF!                 !      !                    !
  3C FF!                 !      !                    !
  3D FF!                 !      !                    !
  3E FF!                 !      !                    !
  3F FF!                 !      !                    !
```

Figure 4-17. Program Listing (Z80)

```
ADRdata!      LABEL        !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0140  46!(INIT SCC CH)     !LDRN  !B,(HL)            !
0141  23!NEXT BYTE         !ICP   !HL                !Output table to channel control
  42  7E!                  !LDAN  !(HL)              !  first selecting the
  43  ED!                  !OPAN  !(C)               !  write register and then
  44  79!                  ! ----                    !  outputting its control byte
  45  05!                  !DCR   !B                 !
  46  C2!                  !JP    !NOT ZERO          !
  47  41!                  ! ----  !NEXT BYTE        !
  48  01!                  ! ----  !                 !
  49  C9!    ----------    !RTS   !UN                !Return
  4A  FF!                  !      !                  !
  4B  FF!                  !      !                  !
  4C  FF!                  !      !                  !
  4D  FF!                  !      !                  !
  4E  FF!                  !      !                  !
  4F  FF!                  !      !                  !
0150  18!ASYNC INIT        !  18  !                  !24(18hex) control bytes
  51  04!                  !  04  !                  !WR4
  52  47!                  !  47  !                  !X16 CLK,1 stop b/c,even parity
  53  01!                  !  01  !                  !WR1
  54  00!                  !  00  !                  !Interrupts disabled
  55  03!                  !  03  !                  !WR3
  56  E0!                  !  E0  !                  !RCV 8b/c,auto enables
  57  05!                  !  05  !                  !WR5
  58  E2!                  !  E2  !                  !XMT 8b/c,DTR* & RTS* enables
  59  0A!                  !  0A  !                  !WR10
  5A  00!                  !  00  !                  !NRZ
  5B  0B!                  !  0B  !                  !WR11
  5C  56!                  !  56  !                  !RCV CLK and XMT CLK =BR gen
  5D  0C!                  !  0C  !                  !WR12 (time constant)
  5E  1E!                  !  1E  !                  !Lower byte  4800 baud
  5F  0D!                  !  0D  !                  !WR13 (time constant)
0160  00!                  !  00  !                  !Upper byte
  61  0E!                  !  0E  !                  !WR14
  62  02!                  !  02  !                  !Baud rate gen source = PCLK
  63  03!                  !  03  !                  !WR3
  64  E1!                  !  E1  !                  !Receiver enable
  65  05!                  !  05  !                  !WR5
  66  EA!                  !  EA  !                  !Transmitter enable
  67  0E!                  !  0E  !                  !WR14
  68  03!                  !  03  !                  !Baud rate generator enable
  69  FF!                  !      !                  !
  6A  FF!                  !      !                  !
  6B  FF!                  !      !                  !
  6C  FF!                  !      !                  !
  6D  FF!                  !      !                  !
  6E  FF!                  !      !                  !
  6F  FF!                  !      !                  !
0170  FF!                  !      !                  !
  71  FF!                  !      !                  !
  72  FF!                  !      !                  !
  73  FF!                  !      !                  !
  74  FF!                  !      !                  !
  75  FF!                  !      !                  !
  76  FF!                  !      !                  !
  77  FF!                  !      !                  !
  78  FF!                  !      !                  !
  79  FF!                  !      !                  !
  7A  FF!                  !      !                  !
  7B  FF!                  !      !                  !
  7C  FF!                  !      !                  !
  7D  FF!                  !      !                  !
  7E  FF!                  !      !                  !
  7F  FF!                  !      !                  !
```

Figure 4-17 (Cont.)

ADR	data	LABEL	INSTR	MODIFIER	COMMENTS
0180	F5	(XMT CH A ASYNC)	PSP	AF	
81	C5		PSP	BC	
82	D5		PSP	DE	
83	E5		PSP	HL	
0184	DB	BUFF EMPTY A	IPA		Wait for end of transmission
85	B1		----	B1 hex	in progress
86	E6		ANAI		
87	04		----	04 hex	
88	CA		JP	ZERO	
89	84		----	BUFF EMPTY A	
8A	01		----		
8B	21		LDPI	HL	Point to data address mailbox
8C	00		----	XMT COUNT A	
8D	05		----		
8E	7E		LDAN	(HL)	Has last character been sent?
8F	FE		CPAI		
0190	00		----	00 hex	
91	CA		JP	ZERO	
92	B0		----	LAST ASYNC	
93	01		----		
94	3D		DCA		No, decrement count and store
95	77		STAN	(HL)	
96	23		ICP	HL	Read address of next character
97	5E		LDRN	E,(HL)	from mailbox
98	23		ICP	HL	
99	56		LDRN	D,(HL)	
9A	1A		LDAN	(DE)	Read next character
9B	13		ICP	DE	Update address of next
9C	72		STRN	D,(HL)	character and store in
9D	2B		DCP	HL	mailbox
9E	73		STRN	E,(HL)	
9F	D3		OPA		Output byte from channel A
01A0	B0		----	B0 hex	
A1	E1		PLP	HL	
A2	D1		PLP	DE	
A3	C1		PLP	BC	
A4	F1		PLP	AF	
A5	C9	----------	RTS	UN	Return
A6	FF				
A7	FF				
A8	FF				
A9	FF				
AA	FF				
AB	FF				
AC	FF				
AD	FF				
AE	FF				
AF	FF				
01B0	D1	LAST ASYNC	PLP	DE	Restore processor registers
B1	F1		PLP	AF	
B2	C3		JP	UN	Go to end point
B3	8F		----	END ASYNC	
B4	03	----------	----		
B5	FF				
B6	FF				
B7	FF				
B8	FF				
B9	FF				
BA	FF				
BB	FF				
BC	FF				
BD	FF				
BE	FF				
BF	FF				

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
01C0  F5!(RCV CH B ASYNC)!PSP   !AF                 !Save processor registers
  C1  E5!                !PSP   !HL                 !
01C2  DB!RECEIVER READY?  !IPA   !                  !Wait until another character
  C3  B3!                !------!B3 hex            !  is received
  C4  E6!                !ANAI  !                   !
  C5  01!                !------!01 hex             !
  C6  CA!                !JP    !ZERO               !
  C7  C2!                !------!RECEIVER READY?    !
  C8  01!                !------!                   !
  C9  21!                !LDPI  !HL                 !Point to channel B status
  CA  05!                !------!ERR STAT B         !  mailbox
  CB  05!                !------!                   !
  CC  3E!                !LDAI  !                   !Store channel B status
  CD  01!                !------!01 hex             !
  CE  D3!                !OPA   !                   !
  CF  B3!                !------!B3 hex             !
01D0  DB!                !IPA   !                   !
  D1  B3!                !------!B3 hex             !
  D2  77!                !STAN  !(HL)               !
  D3  E6!                !ANAI  !                   !If framing error occurs, go to
  D4  40!                !------!40 hex             !  a user-supplied error routine
  D5  C4!                !JS    !NOT ZERO           !
  D6  95!                !------!(FRAMING ERROR)    !
  D7  03!                !------!                   !
  D8  7E!                !LDAN  !(HL)               !
  D9  E6!                !ANAI  !                   !If receiver overrun error
  DA  20!                !------!20 hex             !  occurs, go to user-supplied
  DB  C4!                !JS    !NOT ZERO           !  error routine
  DC  98!                !------!(RCV OVRN ERROR)  !
  DD  03!                !------!                   !
  DE  7E!                !LDAN  !(HL)               !
  DF  E6!                !ANAI  !H,(HL)             !If parity error occurs, go to
01E0  10!                !------!10 hex             !  user-supplied error routine
  E1  C4!                !JS    !NOT ZERO           !
  E2  9B!                !------!(PARITY ERROR)     !
  E3  03!                !------!                   !
  E4  2A!                !LDPD  !HL                 !Get current character
  E5  03!                !------!RCV BUF B          !  store address
  E6  05!                !------!                   !
  E7  DB!                !IPA   !                   !Read character from SCC
  E8  B2!                !------!B2 hex             !  channel B data port
  E9  77!                !STAN  !(HL)               !Store data in memory
  EA  23!                !ICP   !HL                 !Update current character
  EB  22!                !STPD  !HL                 !  store address
  EC  03!                !------!RCV BUF B          !
  ED  05!                !------!                   !
  EE  FE!                !CPAI  !                   !0A = ASCII new line character
  EF  0A!                !------!0A hex             !
01F0  CD!                !JS    !UN                 !When end of file is received
  F1  92!                !------!(PROCESS DATA)     !  process data in user-supplied
  F2  03!                !------!                   !  subroutine
  F3  3E!                !LDAI  !                   !Reset error code
  F4  30!                !------!30 hex             !
  F5  D3!                !OPA   !                   !
  F6  B3!                !------!B3 hex             !
  F7  3E!                !LDAI  !                   !
  F8  33!                !------!33 hex             !
  F9  D3!                !OPA   !                   !
  FA  B3!                !------!B3 hex             !
  FB  E1!                !PLP   !HL                 !Restore processor registers
  FC  F1!                !PLP   !AF                 !
  FD  C9!    ----------  !RTS   !UN                 !
  FE  FF!                !      !                   !
  FF  FF!                !      !                   !
```

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !    MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0200 F5!(INIT SDLC COMM)!PSP   !AF                !Save processor register
  01 C5!                 !PSP   !BC                !
  02 E5!                 !PSP   !HL                !
  03 CD!                 !JS    !UN                !Initialize SCC channels C
  04 10!                 ! ---- !(INIT SCC C)      !  and D for SDLC communication
  05 02!                 ! ---- !                  !
  06 CD!                 !JS    !UN                !
  07 2A!                 ! ---- !(INIT SCC D)      !
  08 02!                 ! ---- !                  !
  09 CD!                 !JS    !UN                !Initialize interrupt structure
  0A C0!                 ! ---- !(INIT INTER)      !
  0B 02!                 ! ---- !                  !
  0C E1!                 !PLP   !HL                !Restore processor registers
  0D C1!                 !PLP   !BC                !
  0E F1!                 !PLP   !AF                !
  0F C9!    ----------   !RTS   !UN                !Return
0210 3E!(INIT SCC C)     !LDAI  !                  !Reset channel C
  11 09!                 ! ---- !09 hex            !
  12 D3!                 !OPA   !                  !
  13 B5!                 ! ---- !B5 hex            !
  14 3E!                 !LDAI  !                  !
  15 80!                 ! ---- !80 hex            !
  16 D3!                 !OPA   !                  !
  17 B5!                 ! ---- !B5 hex            !
  18 21!                 !LDPI  !HL                !Initialize channel C for
  19 40!                 ! ---- !SDLC INIT CH C    !  SDLC communications
  1A 02!                 ! ---- !                  !
  1B 0E!                 !LDRI  !C                 !Channel C control port
  1C B5!                 ! ---- !B5 hex            !  address = B5
  1D CD!                 !JS    !UN                !
  1E 40!                 ! ---- !(INIT SCC CH)     !
  1F 01!                 ! ---- !                  !
0220 C9!    ----------   !RTS   !UN                !Return
  21 FF!                 !      !                  !
  22 FF!                 !      !                  !
  23 FF!                 !      !                  !
  24 FF!                 !      !                  !
  25 FF!                 !      !                  !
  26 FF!                 !      !                  !
  27 FF!                 !      !                  !
  28 FF!                 !      !                  !
  29 FF!                 !      !                  !
022A 3E!(INIT SCC D)     !LDAI  !                  !Reset channel D
  2B 09!                 ! ---- !09 hex            !
  2C D3!                 !OPA   !                  !
  2D B7!                 ! ---- !B7 hex            !
  2E 3E!                 !LDAI  !                  !
  2F 40!                 ! ---- !40 hex            !
0230 D3!                 !OPA   !                  !
  31 B7!                 ! ---- !B7 hex            !
  32 21!                 !LDPI  !HL                !Initialize channel D for
  33 80!                 ! ---- !SDLC INIT CH D    !  SDLC communications
  34 02!                 ! ---- !                  !
  35 0E!                 !LDRI  !C                 !Channel D control port
  36 B7!                 ! ---- !B7 hex            !  address = B7
  37 CD!                 !JS    !UN                !
  38 40!                 ! ---- !(INIT SCC CH)     !
  39 01!                 ! ---- !                  !
  3A C9!    ----------   !RTS   !UN                !Return
  3B FF!                 !      !                  !
  3C FF!                 !      !                  !
  3D FF!                 !      !                  !
  3E FF!                 !      !                  !
  3F FF!                 !      !                  !
```

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !    MODIFIER    !            COMMENTS
-------------------------------------------------------------------------------
0240  25!SDLC INIT CH C   !  25   !                !37(25 hex)control bytes
  41  04!                 !  04   !                !WR4C
  42  20!                 !  20   !                !X1 CLK, SDLC
  43  02!                 !  02   !                !WR2C
  44  10!                 !  10   !                !Interrupt vector
  45  03!                 !  03   !                !WR3C
  46  CC!                 !  CC   !                !RCV 8b/c,RCV CRC en,adr srch
  47  05!                 !  05   !                !WR5C
  48  61!                 !  61   !                !XMT 8b/c,SDLC CRC,XMT CRC en
  49  06!                 !  06   !                !WR6C
  4A  CC!                 !  CC   !                !Ch C SDLC addr (user-selected)
  4B  07!                 !  07   !                !WR7C
  4C  7E!                 !  7E   !                !SDLC flag
  4D  09!                 !  09   !                !WR9C
  4E  01!                 !  01   !                !Vector includes status low
  4F  0A!                 !  0A   !                !WR10C
0250  80!                 !  80   !                !CRC prst=1,NRZ,flag idle&undrrr
  51  0B!                 !  0B   !                !WR11C
  52  16!                 !  16   !                !XMT CLK=BR gen,RCV CLK=RCLK pir
  53  0C!                 !  0C   !                !WR12C (time constant)
  54  7E!                 !  7E   !                !Lower byte 19.2kbaud
  55  0D!                 !  0D   !                !WR13C
  56  00!                 !  00   !                !Upper byte
  57  0E!                 !  0E   !                !WR14C
  58  02!                 !  02   !                !Baud rate gen source = PCLK
  59  03!                 !  03   !                !WR3C
  5A  CD!                 !  CD   !                !Receiver enable
  5B  05!                 !  05   !                !WR5C
  5C  69!                 !  69   !                !Transmitter enable
  5D  80!                 !  80   !                !Reset XMT CRC generator
  5E  0E!                 !  0E   !                !WR14C
  5F  03!                 !  03   !                !Baud rate generator enable
0260  0F!                 !  0F   !                !WR15C
  61  40!                 !  40   !                !Enable XMT underrun/EOM int
  62  10!                 !  10   !                !Reset EXT/STAT interrupts
  63  10!                 !  10   !                !
  64  01!                 !  01   !                !WR1C
  65  12!                 !  12   !                !RCV & XMT interrupt enables
  66  FF!                 !       !                !
  67  FF!                 !       !                !
  68  FF!                 !       !                !
  69  FF!                 !       !                !
  6A  FF!                 !       !                !
  6B  FF!                 !       !                !
  6C  FF!                 !       !                !
  6D  FF!                 !       !                !
  6E  FF!                 !       !                !
  6F  FF!                 !       !                !
0270  FF!                 !       !                !
  71  FF!                 !       !                !
  72  FF!                 !       !                !
  73  FF!                 !       !                !
  74  FF!                 !       !                !
  75  FF!                 !       !                !
  76  FF!                 !       !                !
  77  FF!                 !       !                !
  78  FF!                 !       !                !
  79  FF!                 !       !                !
  7A  FF!                 !       !                !
  7B  FF!                 !       !                !
  7C  FF!                 !       !                !
  7D  FF!                 !       !                !
  7E  FF!                 !       !                !
  7F  FF!                 !       !                !
```

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0280  23!SDLC INIT CH D   !   23  !                  !35(23hex) control bytes
  81  04!                 !   04  !                  !WR4D
  82  20!                 !   20  !                  !X1 CLK,SDLC
  83  03!                 !   03  !                  !WR3D
  84  CC!                 !   CC  !                  !RCV 8b/c,RCV CRC en,adr srch
  85  05!                 !   05  !                  !WR5D
  86  61!                 !   61  !                  !XMT 8b/c,SDLC CRC,XMT CRC en
  87  06!                 !   06  !                  !WR6D
  88  DD!                 !   DD  !                  !Ch D SDLC addr (user-selected)
  89  07!                 !   07  !                  !WR7D
  8A  7E!                 !   7E  !                  !SDLC flag
  8B  0A!                 !   0A  !                  !WR10D
  8C  80!                 !   80  !                  !CRC prst=1,NRZ,flag idle&undrn
  8D  0B!                 !   0B  !                  !WR11D
  8E  16!                 !   16  !                  !XMT CLK=BR gen,RCV CLK=RCLK pi
  8F  0C!                 !   0C  !                  !WR12D (time constant)
0290  7E!                 !   7E  !                  !Lower byte 19.2kbaud
  91  0D!                 !   0D  !                  !WR13D
  92  00!                 !   00  !                  !Upper byte
  93  0E!                 !   0E  !                  !WR14D
  94  02!                 !   02  !                  !Baud rate gen source = PCLK
  95  03!                 !   03  !                  !WR3D
  96  CD!                 !   CD  !                  !Receiver enable
  97  05!                 !   05  !                  !WR5D
  98  69!                 !   69  !                  !Transmitter enable
  99  80!                 !   80  !                  !Reset XMT CRC generator
  9A  0E!                 !   0E  !                  !WR14D
  9B  03!                 !   03  !                  !Baud rate generator enable
  9C  0F!                 !   0F  !                  !WR15D
  9D  40!                 !   40  !                  !Enable XMT underrun/EOM int
  9E  10!                 !   10  !                  !Reset EXT/STAT interrupts
  9F  10!                 !   10  !                  !
02A0  01!                 !   01  !                  !WR1D
  A1  12!                 !   12  !                  !RCV & XMT interrupt enables
  A2  09!                 !   09  !                  !WR9D
  A3  09!                 !   09  !                  !Master interrupt enable
  A4  FF!                 !       !                  !
  A5  FF!                 !       !                  !
  A6  FF!                 !       !                  !
  A7  FF!                 !       !                  !
  A8  FF!                 !       !                  !
  A9  FF!                 !       !                  !
  AA  FF!                 !       !                  !
  AB  FF!                 !       !                  !
  AC  FF!                 !       !                  !
  AD  FF!                 !       !                  !
  AE  FF!                 !       !                  !
  AF  FF!                 !       !                  !
02B0  FF!                 !       !                  !
  B1  FF!                 !       !                  !
  B2  FF!                 !       !                  !
  B3  FF!                 !       !                  !
  B4  FF!                 !       !                  !
  B5  FF!                 !       !                  !
  B6  FF!                 !       !                  !
  B7  FF!                 !       !                  !
  B8  FF!                 !       !                  !
  B9  FF!                 !       !                  !
  BA  FF!                 !       !                  !
  BB  FF!                 !       !                  !
  BC  FF!                 !       !                  !
  BD  FF!                 !       !                  !
  BE  FF!                 !       !                  !
  BF  FF!                 !       !                  !
```

Figure 4-17 (Cont.)

ADR	data	LABEL	INSTR	MODIFIER	COMMENTS
02C0	F5	(INIT INTER)	PSP	AF	Save processor registers
C1	ED		SEIM	2	Select mode 2 interrupts
C2	5E		----		
C3	3E		LDAI		Specify interrupt vector table page address
C4	FF		----	FF hex	
C5	ED		LDR	I,A	
C6	47		----		
C7	FB		ENI		Enable interrupts
C8	F1		PLP	AF	
C9	C9	----------	RTS	UN	Return
CA	FF				
CB	FF				
CC	FF				
CD	FF				
CE	FF				
CF	FF				
02D0	FF				
D1	FF				
D2	FF				
D3	FF				
D4	FF				
D5	FF				
D6	FF				
D7	FF				
D8	FF				
D9	FF				
DA	FF				
DB	FF				
DC	FF				
DD	FF				
DE	FF				
DF	FF				
02E0	F5	(XMT CH C SDLC)	PSP	AF	Save processor registers
02E1	DB	BUFF EMPTY C	IPA		Wait for end of transmission in progress
E2	B5		----	B5 hex	
E3	E6		ANAI		
E4	04		----	04 hex	
E5	CA		JP	ZERO	
E6	E1		----	BUFF EMPTY C	
E7	02		----		
E8	3E		LDAI		Reset XMT CRC generator
E9	80		----	80 hex	
EA	D3		OPA		
EB	B5		----	B5 hex	
EC	3E		LDAI		Transmit secondary address byte (user selected)
ED	DD		----	DD hex	
EE	D3		OPA		
EF	B4		----	B4 hex	
02F0	3E		LDAI		Reset overrun/end of message (OVRN/EOM) latch
F1	C0		----	C0 hex	
F2	D3		OPA		
F3	B5		----	B5 hex	
F4	F1		PLP	AF	Restore processor registers
F5	C9	----------	RTS	UN	Return
F6	FF				
F7	FF				
F8	FF				
F9	FF				
FA	FF				
FB	FF				
FC	FF				
FD	FF				
FE	FF				
FF	FF				

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0300 FB!(XMT C INT)       !ENI    !                  !Enable interrupts
  01 F5!                  !PSP    !AF                !Save processor registers
  02 D5!                  !PSP    !DE                !
  03 E5!                  !PSP    !HL                !
  04 21!                  !LDPI   !HL                !Point to data count
  05 06!                  ! ----  !XMT COUNT C       !  mailbox
  06 05!                  ! ----  !                  !
  07 7E!                  !LDAN   !(HL)              !Has last character been sent?
  08 FE!                  !CPAI   !                  !
  09 00!                  ! ----  !00 hex            !
  0A CA!                  !JP     !ZERO              !
  0B 30!                  ! ----  !LAST SDLC         !
  0C 03!                  ! ----  !                  !
  0D 3D!                  !DCA    !                  !No, decrement count and store
  0E 77!                  !STAN   !(HL)              !
  0F 23!                  !ICP    !HL                !Read address of next character
0310 5E!                  !LDRN   !E,(HL)            !  from mailbox
  11 23!                  !ICP    !HL                !
  12 56!                  !LDRN   !D,(HL)            !
  13 1A!                  !LDAN   !(DE)              !Read next character
  14 13!                  !ICP    !DE                !Update  address of next
  15 72!                  !STRN   !D,(HL)            !  character and store in
  16 2B!                  !DCP    !HL                !  mailbox
  17 73!                  !STRN   !E,(HL)            !
  18 D3!                  !OPA    !                  !Output a byte from SCC CH C
  19 B4!                  ! ----  !B4 hex            !
  1A 3E!                  !LDAI   !                  !Enable lower priority
  1B 38!                  ! ----  !38 hex            !  interrupts
  1C D3!                  !OPA    !                  !
  1D B5!                  ! ----  !B5 hex            !
  1E D3!                  !OPA    !                  !
  1F B7!                  ! ----  !B7 hex            !
0320 D1!                  !PLP    !DE                !Restore processor registers
  21 F1!                  !PLP    !AF                !
  22 C9!    ----------    !RTS    !UN                !Return
  23 FF!                  !       !                  !
  24 FF!                  !       !                  !
  25 FF!                  !       !                  !
  26 FF!                  !       !                  !
  27 FF!                  !       !                  !
  28 FF!                  !       !                  !
  29 FF!                  !       !                  !
  2A FF!                  !       !                  !
  2B FF!                  !       !                  !
  2C FF!                  !       !                  !
  2D FF!                  !       !                  !
  2E FF!                  !       !                  !
  2F FF!                  !       !                  !
0330 3E!LAST SDLC         !LDAI   !                  !Turn XMT interrupt off after
  31 28!                  ! ----  !28 hex            !  last character has been sent
  32 D3!                  !OPA    !                  !
  33 B5!                  ! ----  !B5 hex            !
  34 3E!                  !LDAI   !                  !Enable lower priority
  35 38!                  ! ----  !38 hex            !  interrupts
  36 D3!    ----------    !OPA    !                  !
  37 B5!                  ! ----  !B5 hex            !
  38 D3!                  !OPA    !                  !
  39 B7!                  ! ----  !B7 hex            !
  3A D1!                  !PLP    !DE                !Restore processor registers
  3B F1!                  !PLP    !AF                !
  3C ED!                  !RTI    !                  !Return
  3D 4D!    ----------    ! ----  !                  !
  3E FF!                  !       !                  !
  3F FF!                  !       !                  !
```

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0340 FB!(RCV CH D SDLC)  !ENI   !                  !Enable interrupts
  41 F5!                 !PSP   !AF                 !Save processor registers
  42 E5!                 !PSP   !HL                 !
  43 2A!                 !LDPD  !HL                 !Get current character
  44 09!                 ! ---- !RCV BUF D          !  store address
  45 05!                 ! ---- !                   !
  46 DB!                 !IPA   !                   !Read character from SCC      4
  47 B6!                 ! ---- !B6 hex             !  channel D data port
  48 77!                 !STAN  !(HL)               !Store data in memory
  49 23!                 !ICP   !HL                 !
  4A 22!                 !STPD  !HL                 !Update current character
  4B 09!                 ! ---- !RCV BUF D          !  store address
  4C 05!                 ! ---- !                   !
  4D 3E!                 !LDAI  !                   !Enable lower priority
  4E 30!                 ! ---- !30 hex             !  interrupts
  4F D3!                 !OPA   !                   !
0350 B7!                 ! ---- !B7 hex             !
  51 E1!                 !PLP   !HL                 !
  52 F1!                 !PLP   !AF                 !Restore processor registers
  53 ED!                 !RTI   !                   !Return
  54 4D!                 ! ---- !                   !
  55 FF!                 !      !                   !
  56 FF!                 !      !                   !
  57 FF!                 !      !                   !
  58 FF!                 !      !                   !
  59 FF!                 !      !                   !
  5A FF!                 !      !                   !
035B FB!(SPCL RCVD SDLC)!ENI   !                   !Enable interrupts
  5C F5!                 !PSP   !AF                 !Save processor registers
  5D E5!                 !PSP   !HL                 !
  5E 21!                 !LDPI  !HL                 !Point to channel D status
  5F 0B!                 ! ---- !ERR STAT D         !  mailbox
0360 05!                 ! ---- !                   !
  61 3E!                 !LDAI  !                   !Read and store channel D
  62 01!                 ! ---- !01 hex             !  status
  63 D3!                 !OPA   !                   !
  64 B7!                 ! ---- !B7 hex             !
  65 DB!                 !IPA   !                   !
  66 B7!                 ! ---- !B7 hex             !
  67 77!                 !STAN  !(HL)               !
  68 E6!                 !ANAI  !                   !If SDLC CRC error occurs,go to
  69 40!                 ! ---- !40 hex             !  user-supplied error routine
  6A C4!                 !JS    !NOT ZERO           !
  6B B6!                 ! ---- !(CRC ERR D)        !
  6C 03!                 ! ---- !                   !
  6D E6!                 !ANAI  !                   !If receiver overrun error
  6E 20!                 ! ---- !20 hex             !  occurs, go to user-supplied
  6F C4!                 !JS    !NOT ZERO           !  error routine
0370 B9!                 ! ---- !(OVRN ERR D)       !
  71 03!                 ! ---- !                   !
  72 7E!                 !LDAN  !(HL)               !
  73 E6!                 !ANAI  !                   !If end of frame is detected,
  74 80!                 ! ---- !80 hex             !  go to user-supplied routine
  75 C4!                 !JS    !NOT ZERO           !
  76 B3!                 ! ---- !(END OF FRAME D)   !
  77 03!                 ! ---- !                   !
  78 3E!                 !LDAI  !                   !Reset error code
  79 30!                 ! ---- !30 hex             !
  7A D3!                 !OPA   !                   !
  7B B7!                 ! ---- !B7 hex             !
  7C E1!                 !PLP   !HL                 !Restore processor registers
  7D F1!                 !PLP   !AF                 !
  7E ED!                 !RTI   !                   !Return
  7F 4D!                 ! ---- !                   !
```

Figure 4-17 (Cont.)

```
ADRdata!      LABEL       !INSTR !     MODIFIER     !            COMMENTS
-------------------------------------------------------------------------------
0380  31!(ASYNC COMM)      !LDPI  !SP                !Load stack pointer
  81  FF!                  ! ---- !adr 00FF          !
  82  00!                  ! ---- !                  !
  83  CD!                  !JS    !UN                !Initialize channels A and B
  84  00!                  ! ---- !(INIT ASYN COMM)  !
  85  01!                  ! ---- !                  !
0386  CD!ASYNC LOOP        !JS    !UN                !Transmit a byte
  87  80!                  ! ---- !(XMT CH A ASYNC)  !
  88  01!                  ! ---- !                  !
  89  CD!                  !JS    !UN                !Receive a byte
  8A  C0!                  ! ---- !(RCV CH B ASYNC)  !
  8B  01!                  ! ---- !                  !
  8C  C3!                  !JP    !UN                !Loop until last byte
  8D  86!                  ! ---- !ASYNC LOOP        !
  8E  03!   ----------     ! ---- !                  !
038F  C3!END ASYNC         !JP    !UN                !End of transmission
0390  8F!                  ! ---- !END ASYNC         !
  91  03!   ----------     ! ---- !                  !
0392  00!(PROCESS DATA)    !NOP   !                  !USER-SUPPLIED SUBROUTINE
  93  00!                  !NOP   !                  !
  94  C9!   ----------     !RTS   !UN                !
0395  00!(FRAMING ERROR)   !NOP   !                  !USER-SUPPLIED SUBROUTINE
  96  00!                  !NOP   !                  !
  97  C9!   ----------     !RTS   !UN                !
0398  00!(RCV OVRN ERROR)  !NOP   !                  !USER-SUPPLIED SUBROUTINE
  99  00!                  !NOP   !                  !
  9A  C9!   ----------     !RTS   !UN                !
039B  00!(PARITY ERROR)    !NOP   !                  !USER-SUPPLIED SUBROUTINE
  9C  00!                  !NOP   !                  !
  9D  C9!   ----------     !RTS   !UN                !
  9E  FF!                  !      !                  !
  9F  FF!                  !      !                  !
03A0  31!(SDLC COMM)       !LDPI  !SP                !Load stack pointer
  A1  FF!                  ! ---- !adr 00FF          !
  A2  00!                  ! ---- !                  !
  A3  CD!                  !JS    !UN                !Initialize channels C and D
  A4  00!                  ! ---- !(INIT SDLC COMM)  !
  A5  02!                  ! ---- !                  !
  A6  CD!                  !JS    !UN                !Transmit first byte
  A7  E0!                  ! ---- !(XMT CH C SDLC)   !
  A8  02!                  ! ---- !                  !
03A9  C2!WAIT              !JP    !NOT ZERO          !Wait for transmission of
  AA  A9!                  ! ---- !WAIT              !  all characters in block
  AB  03!                  ! ---- !                  !
  AC  FF!                  !      !                  !
  AD  FF!                  !      !                  !
  AE  FF!                  !      !                  !
  AF  FF!                  !      !                  !
03B0  C3!END SDLC          !JP    !UN                !End of transmission
  B1  B0!                  ! ---- !END SDLC          !
  B2  03!   ----------     ! ---- !                  !
03B3  00!(END OF FRAME D)  !NOP   !                  !USER-SUPPLIED SUBROUTINE
  B4  00!                  !NOP   !                  !
  B5  C9!   ----------     !RTS   !UN                !
03B6  00!(CRC ERR D)       !NOP   !                  !USER-SUPPLIED SUBROUTINE
  B7  00!                  !NOP   !                  !
  B8  C9!   ----------     !RTS   !UN                !
03B9  00!(OVRN ERR D)      !NOP   !                  !USER-SUPPLIED SUBROUITNE
  BA  00!                  !NOP   !                  !
  BB  C9!   ----------     !RTS   !UN                !
  BC  FF!                  !      !                  !
  BD  FF!                  !      !                  !
  BE  FF!                  !      !                  !
  BF  FF!                  !      !                  !
```

Figure 4-17 (Cont.)

ADR	data	LABEL	INSTR	MODIFIER	COMMENTS
0500	FF	XMT COUNT A			Number of CH A XMT characters
0501	FF	XMT BUF A			Address of current CH A
02	FF				transmit character
0503	FF	RCV BUF B			Address of current CH B
04	FF				receive character
0505	FF	ERR STAT B			Channel B error status
0506	FF	XMT COUNT C			Number of CH C XMT characters
0507	FF	XMT BUF C			Address of current CH C
08	FF				transmit character
0509	FF	RCV BUF D			Address of current CH D
0A	FF				receiiver character
050B	FF	ERR STAT D			Channel D error status
0C	FF				
0D	FF				
0E	FF				
0F	FF				
0510	FF				
11	FF				
12	FF				
13	FF				
14	FF				
15	FF				
16	FF				
17	FF				
18	FF				
19	FF				
1A	FF				
1B	FF				
1C	FF				
1D	FF				
1E	FF				
1F	FF				
0520	FF				
21	FF				
22	FF				
23	FF				
24	FF				
25	FF				
26	FF				
27	FF				
28	FF				
29	FF				
2A	FF				
2B	FF				
2C	FF				
2D	FF				
2E	FF				
2F	FF				
0530	FF				
31	FF				
32	FF				
33	FF				
34	FF				
35	FF				
36	FF				
37	FF				
38	FF				
39	FF				
3A	FF				
3B	FF				
3C	FF				
3D	FF				
3E	FF				
3F	FF				

Figure 4-17 (Cont.)

ADR	data	LABEL	INSTR	MODIFIER	COMMENTS
FF00	FF	VECTOR TABLE			Channel B XMT buffer empty
01	FF				
02	FF				Channel B external/status change
03	FF				
04	FF				Channel B RCV character available
05	FF				
06	FF				Channel B special RCV
07	FF				
08	FF				Channel A XMT buffer empty
09	FF				
0A	FF				Channel A external/status change
0B	FF				
0C	FF				Channel A RCV character available
0D	FF				
0E	FF				Channel A special RCV
0F	FF				
FF10	FF				Channel D XMT buffer empty
11	FF				
12	FF				Channel D external/status change
13	FF				
14	40		40		Channel D RCV character available
15	03		03		
16	5B		5B		Channel D special RCV
17	03		03		
18	00		00		Channel C XMT buffer empty
19	03		03		
1A	FF				Channel C external/status change
1B	FF				
1C	FF				Channel C RCV character available
1D	FF				
1E	FF				Channel C special RCV
1F	FF				
FF20	FF				
21	FF				
22	FF				
23	FF				
24	FF				
25	FF				
26	FF				
27	FF				
28	FF				
29	FF				
2A	FF				
2B	FF				
2C	FF				
2D	FF				
2E	FF				
2F	FF				
FF30	FF				
31	FF				
32	FF				
33	FF				
34	FF				
35	FF				
36	FF				
37	FF				
38	FF				
39	FF				
3A	FF				
3B	FF				
3C	FF				
3D	FF				
3E	FF				
3F	FF				

Figure 4-17 (Cont.)

```
The IBM Personal Computer MACRO Assembler 07-11-85
1                                       PAGE    ,132
2
3                                       ;7314A/7312 Channel A and Channel B SCC Asynchrous Drivers
4
5
6                                       ;*****************************************************************************
7                                       ;* These are the 7314A/7312 Channel A and Channel B port addresses.          *
8                                       ;*****************************************************************************
9       = 00B0                          SCC_DTA_A       EQU     000B0H
10      = 00B1                          SCC_CMD_A       EQU     000B1H
11      = 00B2                          SCC_DTA_B       EQU     000B2H
12      = 00B3                          SCC_CMD_B       EQU     000B3H
13
14
15
16
17                                      ;*****************************************************************************
18                                      ;* The following memory locations are used as mailboxes.  Transmit and receive buffer    *
19                                      ;* pointers, error status bytes, and transmit byte counts are stored here.          *
20                                      ;*****************************************************************************
21      0000                            MAILBOX SEGMENT AT 0100H
22      0000  ????                      XMT_COUNT_A     DW      ?               ;Transmit byte count CH A
23      0002  ????????                  XMT_BUF_A       DD      ?               ;Transmit buffer pointer CH A
24      0006  ????????                  RCV_BUF_B       DD      ?               ;Receive buffer pointer CH B
25      000A  ??                        ERR_STAT_B      DB      ?               ;Error status CH B
26      000B                            MAILBOX ENDS
27
28      0000                            BUFFER  SEGMENT AT 0
29      0000  ??                        XMT_BUFFER      DB      ?               ;Arbitrary value defined by XMT_BUF_A
30      0001  ??                        RCV_BUFFER      DB      ?               ;Arbitrary value defined by RCV_BUF_B
31      0002                            BUFFER  ENDS
32
33
34
35
36
37
38
39
40
41
42      0000                            CODE            SEGMENT PUBLIC
43                                                      ASSUME  CS:CODE
44
45
46
47
48      0000                            TEST_ASYNC      PROC    FAR             ;Go to main subroutine
49      0000  E9 00FD R                                 JMP     ASYNC_COMM
50      0003                            TEST_ASYNC      ENDP
51
52
53
```

Figure 4-18. Program Listing (8088)

```
                                ;*****************************************************************************
                                ;* INIT_SCC_A                                                                *
                                ;*       This subroutine initializes Channel A of the SCC with the data in the table  *
                                ;*       starting at ASYNC_INIT.                                             *
                                ;*                                                                           *
                                ;* Entry:  None                                                              *
                                ;* Exit:   Channel A of the SCC initialized                                  *
                                ;* Alters: AL, Flags                                                         *
                                ;* Calls:  None                                                              *
                                ;* Stack:  5 words                                                           *
                                ;*****************************************************************************
                                ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0003                            INIT_SCC_A      PROC    FAR
0003  51                                        PUSH    CX                              ;Save registers
0004  52                                        PUSH    DX
0005  56                                        PUSH    SI
0006  EB 18 90                                 JMP     INIT_A                          ;Jump past the table
0009  18                        ASYNC_INIT      DB      OFFSET ASYNC_INIT_END - OFFSET ASYNC_INIT-1
000A  04 47                                     DB      04,047H         ;Register 4, x16 Clk/1Stop/Even Parity
000C  03 E0                                     DB      03,0E0H         ;Register 3, Rcv 8 Bits, Auto Enables
000E  05 E2                                     DB      05,0E2H         ;Register 5, Xmt 8 Bits, DTR and RTS enabled
0010  0A 00                                     DB      10,000H         ;Register 10, Misc Control
0012  0B 56                                     DB      11,056H         ;Register 11, Rcv Clk=BR Gen, Xmt Clk=BR Gen
0014  0C 1E                                     DB      12,01EH         ;Register 12, Low  Time Const. Byte (4800)
0016  0D 00                                     DB      13,000H         ;Register 13, High Time Const. Byte (4800)
0018  0E 02                                     DB      14,002H         ;Register 14, Baud Rate Gen=PCLK Input
001A  03 E1                                     DB      03,0E1H         ;Register 3, Rcv 8 Bits, Rcv Enabled
001C  05 EA                                     DB      05,0EAH         ;Register 5, En DTR, En RTS, Xmt 8 BITS, En Xmt
001E  0E 03                                     DB      14,003H         ;Register 14, BR Gen=PCLK Input, En BR Gen
0020                            ASYNC_INIT_END LABEL    BYTE            ;End of table

0020  BA 00B1                   INIT_A:         MOV     DX,SCC_CMD_A                    ;Port addr to DX
0023  B0 09                                     MOV     AL,09H                          ;Reset channel A
0025  EE                                        OUT     DX,AL
0026  B0 80                                     MOV     AL,80H
0028  EE                                        OUT     DX,AL
0029  BE 0009 R                                 MOV     SI,OFFSET ASYNC_INIT            ;Get table address
002C  2E: AC                                    LODS    ASYNC_INIT                      ;Byte count to CX
002E  8A C8                                     MOV     CL,AL
0030  32 ED                                     XOR     CH,CH
0032  BA 00B1                                   MOV     DX,SCC_CMD_A                    ;Port addr to DX
0035                            INIT_A_LP:
0035  2E: AC                                    LODS    ASYNC_INIT                      ;Get byte
0037  EE                                        OUT     DX,AL                           ;Output to SCC CH A
0038  E2 FB                                     LOOP    INIT_A_LP                       ;Repeat
003A  5E                                        POP     SI                              ;Restore registers
003B  5A                                        POP     DX
003C  59                                        POP     CX
003D  CB                                        RET                                     ;Return
003E                            INIT_SCC_A      ENDP
```

```
                                ;*****************************************************************************
                                ;* INIT_SCC_B                                                                *
                                ;*       This subroutine initializes Channel B of the SCC with the data in the table  *
                                ;*       starting at ASYNC_INIT.                                             *
                                ;*                                                                           *
                                ;* Entry:  None                                                              *
                                ;* Exit:   Channel B of the SCC initialized.                                 *
                                ;* Alters: AL, Flags                                                         *
                                ;* Calls:  None                                                              *
                                ;* Stack:  5 words                                                           *
                                ;*****************************************************************************
                                ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
003E                            INIT_SCC_B      PROC    FAR
003E  51                                        PUSH    CX                      ;Save registers
003F  52                                        PUSH    DX
0040  56                                        PUSH    SI
0041  BA 00B3                  INIT_B:         MOV     DX,SCC_CMD_B            ;Port addr to DX
0044  B0 09                                     MOV     AL,09H                  ;Reset channel B
0046  EE                                        OUT     DX,AL
0047  B0 40                                     MOV     AL,40H
0049  EE                                        OUT     DX,AL
004A  BE 0009 R                                 MOV     SI,OFFSET ASYNC_INIT    ;Get table address
004D  2E: AC                                    LODS    ASYNC_INIT              ;Byte count to CX
004F  8A C8                                     MOV     CL,AL
0051  32 ED                                     XOR     CH,CH
0053  BA 00B3                                  MOV     DX,SCC_CMD_B            ;Port addr to DX
0056                            INIT_B_LP:
0056  2E: AC                                    LODS    ASYNC_INIT              ;Get byte
0058  EE                                        OUT     DX,AL                   ;Output to SCC CH B
0059  E2 FB                                     LOOP    INIT_B_LP               ;Repeat
005B  5E                                        POP     SI                      ;Restore registers
005C  5A                                        POP     DX
005D  59                                        POP     CX
005E  CB                                        RET                             ;Return
005F                            INIT_SCC_B      ENDP
```

```
                        ;*****************************************************************************
                        ;* XMT_CH_A_ASYN                                                             *
                        ;*       This subroutine asynchronously transmits a block of data of up to 255 bytes.  *
                        ;*       The program first waits for any previous transmission to be completed.  *
                        ;*       The first byte is sent and the program exits.                       *
                        ;*       When the last byte is sent the program exits to LAST_ASYNC.         *
                        ;*       The XMT_COUNT_A mailbox must contain the byte count,                *
                        ;*       and the XMT_BUF_A mailbox must contain the address of the first byte  *
                        ;*       to be sent before this subroutine is executed.                      *
                        ;*                                                                           *
                        ;*                                                                           *
                        ;* Entry:  XMT_COUNT_A: Number of bytes to be sent                           *
                        ;*         XMT_BUF_A: Address of first byte to be sent                       *
                        ;*         Memory: Data to be sent                                           *
                        ;* Exit:   Byte count decremented and stored at (XMT_COUNT_A)                *
                        ;*         Data pointer incremented and stored at (XMT_BUF_A)                *
                        ;*         One byte sent                                                     *
                        ;* Alters: AL, Flags                                                         *
                        ;* Calls:  None                                                              *
                        ;* Stack:  7 words                                                           *
                        ;*****************************************************************************
                        ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
005F                    XMT_CH_A_ASYNC  PROC    FAR
005F  50                                PUSH    AX                      ;Save registers
0060  1E                                PUSH    DS
0061  56                                PUSH    SI
0062  52                                PUSH    DX
0063  06                                PUSH    ES
0064  BA 00B1                           MOV     DX,SCC_CMD_A            ;Point to CH A command port
0067  EC                  BUFMPTY_A:    IN      AL,DX                   ;Wait for Xmt Buf empty
0068  24 04                             AND     AL,04H
006A  3C 04                             CMP     AL,04H
006C  75 F9                             JNE     BUFMPTY_A
006E  B8 ---- R                         MOV     AX,SEG MAILBOX
0071  8E D8                             MOV     DS,AX
                                        ASSUME  DS:MAILBOX
0073  83 3E 0000 R 00                   CMP     XMT_COUNT_A,0           ;Last character sent?
0078  74 13                             JE      LAST_ASYNC              ;Yes, exit subroutine
007A  FF 0E 0000 R                      DEC     XMT_COUNT_A             ;No, decrement byte count
007E  C4 36 0002 R                      LES     SI,XMT_BUF_A            ;Point to XMT buffer A
                                        ASSUME  ES:BUFFER
0082  FC                                CLD
0083  26: AC                           LODS    XMT_BUFFER
0085  BA 00B0                          MOV     DX,SCC_DTA_A
0088  89 36 0002 R                      MOV     WORD PTR XMT_BUF_A,SI
008C  EE                                OUT     DX,AL
008D  07                  LAST_ASYNC:   POP     ES                      ;Restore registers
008E  5A                                POP     DX
008F  5E                                POP     SI
0090  1F                                POP     DS
0091  58                                POP     AX
0092  CB                                RET                             ;Return
0093                      XMT_CH_A_ASYNC  ENDP
```

```
                        ;*****************************************************************************
                        ;* PROCESS_DATA                                                              *
                        ;*       The user must supply this subroutine, which is called when a new line *
                        ;*       character is received by Channel B, indicating the end of an        *
                        ;*       asynchronous message.                                               *
                        ;*****************************************************************************
                        ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0093                    PROCESS_DATA    PROC FAR
0093 CB                                 RET
0094                    PROCESS_DATA    ENDP


                        ;*****************************************************************************
                        ;* FRAME_ERR_B, OVRN_ERR_B, PAR_ERR_B (All user-supplied)                    *
                        ;*       These subroutines are called by the RCV_CH_B_ASYNC                  *
                        ;*       routine in response to various error conditions.                    *
                        ;*****************************************************************************
                        ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING

0094                    FRAME_ERR_B     PROC    FAR
0094 CB                                 RET
0095                    FRAME_ERR_B     ENDP

0095                    OVRN_ERR_B      PROC    FAR
0095 CB                                 RET
0096                    OVRN_ERR_B      ENDP

0096                    PAR_ERR_B       PROC    FAR
0096 CB                                 RET
0097                    PAR_ERR_B       ENDP
```

```
                        ;*****************************************************************************
                        ;* RCV_CH_B_ASYNC                                                            *
                        ;*       This routine receives an asynchronously transmitted character and   *
                        ;*       stores it in memory.  The user must supply a subroutine (PROCESS_DATA) which *
                        ;*       interprets the received data string when a new line character is received. *
                        ;*       If a framing error, receiver overrun, or parity error occurs,       *
                        ;*       the program jumps to a user-supplied error routine.                 *
                        ;*                                                                           *
                        ;* Entry:  None                                                              *
                        ;* Exit:   Character stored in buffer                                        *
                        ;*       Next storage address stored at (RCV_BUF_B).                         *
                        ;* Alters: None                                                              *
                        ;* Calls:  PROCESS_DATA (user-supplied)                                      *
                        ;* Stack:  7 words                                                           *
                        ;*****************************************************************************
                        ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0097                    RCV_CH_B_ASYNC  PROC    FAR
0097  50                                PUSH    AX                  ;Save registers
0098  52                                PUSH    DX
0099  57                                PUSH    DI
009A  BA 00B3                           MOV     DX,SCC_CMD_B
009D  EC                REC_READY?:     IN      AL,DX               ;Wait for a receive character
009E  24 01                             AND     AL,01H
00A0  3C 01                             CMP     AL,01H
00A2  75 F9                             JNE     REC_READY?
00A4  B8  ---- R                        MOV     AX,SEG MAILBOX
00A7  8E D8                             MOV     DS,AX
                                        ASSUME  DS:MAILBOX
00A9  BA 00B3                           MOV     DX,SCC_CMD_B        ;Point to SCC CH B, RR1
00AC  B0 01                             MOV     AL,001H
00AE  EE                                OUT     DX,AL
00AF  EC                                IN      AL,DX               ;Read error status
00B0  A2 000A R                         MOV     ERR_STAT_B,AL       ;Store it
00B3  24 40                             AND     AL,040H             ;Framing error ?
00B5  74 05                             JZ      GO_ON3
00B7  9A 0094 ---- R                    CALL    FRAME_ERR_B         ;Yes
00BC  A0 000A R                 GO_ON3: MOV     AL,ERR_STAT_B       ;Overrun error ?
00BF  24 20                             AND     AL,020H
00C1  74 05                             JZ      GO_ON4
00C3  9A 0095 ---- R                    CALL    OVRN_ERR_B          ;Yes
00C8  A0 000A R                 GO_ON4: MOV     AL,ERR_STAT_B       ;Parity Error ?
00CB  24 80                             AND     AL,080H
00CD  74 05                             JZ      GO_ON5
00CF  9A 0096 ---- R                    CALL    PAR_ERR_B           ;Yes
00D4  B0 30                     GO_ON5: MOV     AL,030H             ;Error reset
00D6  EE                                OUT     DX,AL
```

```
00D7  BA 00B2                          MOV     DX,SCC_DTA_B            ;Point to CH B data port
00DA  C4 3E 0006 R                     LES     DI,RCV_BUF_B            ;Point ES:DI to CH B Rcv Buffer
                                       ASSUME  ES:BUFFER
00DE  EC                               IN      AL,DX                   ;Input data
00DF  FC                               CLD
00E0  AA                               STOS    RCV_BUFFER              ;Store received data in buffer
00E1  89 3E 0006 R                     MOV     RCV_BUF_B,DI            ;Store next buffer location
00E5  3C 0A                            CMP     AL,0AH                  ;ASCII new line character?
00E7  75 05                            JNE     CONTINUE                ;No - continue
00E9  9A 0093 ---- R                   CALL    PROCESS_DATA            ;End of message, go process data
00EE  5F                       CONTINUE:       POP     DI              ;Restore registers
00EF  5A                               POP     DX
00F0  58                               POP     AX
00F1  CB                               RET                             ;Return
00F2                           RCV_CH_B_ASYNC  ENDP
```

```
                               ;*****************************************************************************
                               ;* INIT_ASYN_COMM                                                            *
                               ;*       This subroutine initializes SCC Channels A and B.  The initialized conditions *
                               ;*       should be checked to be sure they match the application.  The baud rates for  *
                               ;*       Channel A and Channel B are 4800, using the asynchronous protocol.          *
                               ;*                                                                           *
                               ;* Entry:  None                                                              *
                               ;* Exit:   SCC initialized.                                                  *
                               ;* Alters: AX, DX, DI, Flags                                                 *
                               ;* Calls:  INIT_SCC_A, INIT_SCC_B                                            *
                               ;* Stack:  12 words                                                          *
                               ;*****************************************************************************
                               ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
00F2                           INIT_ASYN_COMM  PROC    FAR
00F2  9A 0003 ---- R                           CALL    INIT_SCC_A      ;Initialize SCC Channel A
00F7  9A 003E ---- R                           CALL    INIT_SCC_B      ;Initialize SCC Channel B
00FC  CB                                       RET                     ;Return
00FD                           INIT_ASYN_COMM  ENDP
```

```
372
373
374
375
376                                     ;*****************************************************************************
377                                     ;* ASYNC_COMM                                                                *
378                                     ;*       This is a main subroutine which initializes the SCC and transmits  a character  *
379                                     ;*       from Channel A to Channel B.  When all of the characters have been transmitted, *
380                                     ;*       the program exits to END_ASYNC.                                     *
381                                     ;*                                                                           *
382                                     ;* Entry:  Data to be transmitted from Channel A in memory, staring at XMT_BUF_A.  *
383                                     ;* Exit:   SCC initialized, Channel A data transmitted to Channel B          *
384                                     ;* Alters: None                                                              *
385                                     ;* Calls:  INIT_ASYN_COMM, XMT_CH_A_ASYNC, RCV_CH_B_ASYNC                    *
386                                     ;* Stack:  26 words                                                          *
387                                     ;*****************************************************************************
388
389                                     ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
390     00FD                            ASYNC_COMM      PROC    NEAR
391     00FD  9A 00F2 ---- R                            CALL    INIT_ASYN_COMM          ;Initialize SCC channels A and B
392     0102  B8  ---- R                                MOV     AX,SEG MAILBOX
393     0105  8E D8                                     MOV     DS,AX
394                                                     ASSUME  DS:MAILBOX
395     0107  C7 06 0000 R 000F                         MOV     XMT_COUNT_A,0FH
396     010D  C7 06 0002 R 0100                         MOV     WORD PTR XMT_BUF_A,0100H        ;Point to transmit buffer address
397     0113  C7 06 0004 R 0100                         MOV     WORD PTR XMT_BUF_A+2,0100H
398     0119  C7 06 0006 R 0300                         MOV     WORD PTR RCV_BUF_B,0300H        ;Point to receive buffer address
399     011F  C7 06 0008 R 0100                         MOV     WORD PTR RCV_BUF_B+2,0100H
400     0125  9A 005F ---- R            ASYNC_LOOP:     CALL    XMT_CH_A_ASYNC          ;Transmit a character from channel
                                                A
401     012A  9A 0097 ---- R                            CALL    RCV_CH_B_ASYNC          ;Receive a character at channel B
402     012F  83 3E 0000 R 00                           CMP     XMT_COUNT_A,0           ;End of transmission?
403     0134  75 EF                                     JNE     ASYNC_LOOP              ;Loop until last character is sent.
404     0136  EB FE                     END_ASYNC:      JMP     END_ASYNC               ;Final loop
405     0138                            ASYNC_COMM      ENDP
406     0138                            CODE    ENDS
407                                     END
```

```
Segments and groups:

                N a m e                 Size    align   combine class

BUFFER . . . . . .              0002    AT      0000
CODE . . . . . .                0138    PARA    PUBLIC
MAILBOX. . . . . .              000B    AT      0100

Symbols:

                N a m e                 Type    Value   Attr

ASYNC_COMM . . . . . .          N PROC  00FD    CODE    Length =003B
ASYNC_INIT . . . . . .          L BYTE  0009    CODE
ASYNC_INIT_END . . . . . .      L BYTE  0020    CODE
ASYNC_LOOP . . . . . .          L NEAR  0125    CODE
BUFMPTY_A. . . . . .            L NEAR  0067    CODE
CONTINUE . . . . . .            L NEAR  00EE    CODE
END_ASYNC. . . . . .            L NEAR  0136    CODE
ERR_STAT_B . . . . . .          L BYTE  000A    MAILBOX
FRAME_ERR_B. . . . . .          F PROC  0094    CODE    Length =0001
GO_ON3 . . . . . .              L NEAR  00BC    CODE
GO_ON4 . . . . . .              L NEAR  00C8    CODE
GO_ON5 . . . . . .              L NEAR  00D4    CODE
INIT_A . . . . . .              L NEAR  0020    CODE
INIT_ASYN_COMM . . . . . .      F PROC  00F2    CODE    Length =000B
INIT_A_LP. . . . . .            L NEAR  0035    CODE
INIT_B . . . . . .              L NEAR  0041    CODE
INIT_B_LP. . . . . .            L NEAR  0056    CODE
INIT_SCC_A . . . . . .          F PROC  0003    CODE    Length =003B
INIT_SCC_B . . . . . .          F PROC  003E    CODE    Length =0021
LAST_ASYNC . . . . . .          L NEAR  008D    CODE
OVRN_ERR_B . . . . . .          F PROC  0095    CODE    Length =0001
PAR_ERR_B. . . . . .            F PROC  0096    CODE    Length =0001
PROCESS_DATA . . . . . .        F PROC  0093    CODE    Length =0001
RCV_BUFFER . . . . . .          L BYTE  0001    BUFFER
RCV_BUF_B. . . . . .            L DWORD 0006    MAILBOX
RCV_CH_B_ASYNC . . . . . .      F PROC  0097    CODE    Length =005B
REC_READY? . . . . . .          L NEAR  009D    CODE
SCC_CMD_A. . . . . .            Number  00B1
SCC_CMD_B. . . . . .            Number  00B3
SCC_DTA_A. . . . . .            Number  00B0
SCC_DTA_B. . . . . .            Number  00B2
TEST_ASYNC . . . . . .          F PROC  0000    CODE    Length =0003
XMT_BUFFER . . . . . .          L BYTE  0000    BUFFER
XMT_BUF_A. . . . . .            L DWORD 0002    MAILBOX
XMT_CH_A_ASYNC . . . . . .      F PROC  005F    CODE    Length =0034
XMT_COUNT_A. . . . . .          L WORD  0000    MAILBOX

Warning Severe
Errors  Errors
0       0
```

```
1                                       PAGE    ,132
2                                       ;7314A Channel C and Channel D SDLC Drivers
3                                       ;*****************************************************************************
4                                       ;* Interrupt type number = 240 (F0H).                                        *
5                                       ;* The user can specify a different type number if required.                 *
6                                       ;*****************************************************************************
7       = 00F0                          INTVECT         EQU     0F0H
8                                       ;*****************************************************************************
9                                       ;*  These are the 7314A Channel C and Channel D port addresses.              *
10                                      ;*****************************************************************************
11      = 00B4                          SCC_DTA_C       EQU     000B4H
12      = 00B5                          SCC_CMD_C       EQU     000B5H
13      = 00B6                          SCC_DTA_D       EQU     000B6H
14      = 00B7                          SCC_CMD_D       EQU     000B7H
15                                      ;*****************************************************************************
16                                      ;*  This is the vector table for channels C and D of the SCC.                *
17                                      ;*****************************************************************************
18      0000                            VECTOR  SEGMENT AT 000H
19      03C0                            ORG             4*INTVECT
20      03C0  ????????                  INTR0                   DD      ?       ;Channel D transmit buffer empty
21      03C4  ????????                  INTRX0          DD      ?       ;Unused vector
22      03C8  ????????                  INTR1                   DD      ?       ;Channel D status change
23      03CC  ????????                  INTRX1          DD      ?       ;Unused vector
24      03D0  ????????                  INTR2                   DD      ?       ;Channel D receive character available
25      03D4  ????????                  INTRX2          DD      ?       ;Unused vector
26      03D8  ????????                  INTR3                   DD      ?       ;Channel D special receive condition
27      03DC  ????????                  INTRX3          DD      ?       ;Unused vector
28      03E0  ????????                  INTR4                   DD      ?       ;Channel C transmit buffer empty
29      03E4  ????????                  INTRX4          DD      ?       ;Unused vector
30      03E8  ????????                  INTR5                   DD      ?       ;Channel C status change
31      03EC  ????????                  INTRX5          DD      ?       ;Unused vector
32      03F0  ????????                  INTR6                   DD      ?       ;Channel C receive character available
33      03F4  ????????                  INTRX6          DD      ?       ;Unused vector
34      03F8  ????????                  INTR7                   DD      ?       ;Channel C special receive condition
35      03FC  ????????                  INTRX7          DD      ?       ;Unused vector
36      0400                            VECTOR  ENDS
37                                      ;*****************************************************************************
38                                      ;* The following memory locations are used as mailboxes.  Transmit and receive buffer    *
39                                      ;* pointers, error status bytes, and transmit byte counts are stored.        *
40                                      ;*****************************************************************************
41      0000                            MAILBOX SEGMENT AT 0100H
42      0000  ????                              XMT_COUNT_C     DW      ?       ;Channel C transmit byte count
43      0002  ????????                          XMT_BUF_C       DD      ?       ;Channel C transmit buffer pointer
44      0006  ????????                          RCV_BUF_D       DD      ?       ;Channel D receive buffer pointer
45      000A  ??                                ERR_STAT_D      DB      ?       ;Channel D error status
46      000B                            MAILBOX ENDS
47      0000                            BUFFER  SEGMENT AT 0
48      0000  ??                                XMT_BUFFER      DB      ?       ;Arbitrary value defined by XMT_BUF_C
49      0001  ??                                RCV_BUFFER      DB      ?       ;Arbitrary value defined by RCV_BUF_D
50      0002                            BUFFER  ENDS
51
52
53
```

```
54
55      0000                    CODE            SEGMENT PUBLIC
56                                              ASSUME  CS:CODE
57
58
59      0000                    TEST_SDLC       PROC    FAR             ;Go to main subroutine
60      0000  E9 0196 R                         JMP     SDLC_COMM
61      0003                    TEST_SDLC       ENDP
62
63
64
65                              ;*******************************************************************************
66                              ;* INIT_SCC_C                                                                  *
67                              ;*       This subroutine initializes Channel C of the SCC with the data in the table  *
68                              ;*       starting at SDLC_INIT_CH_C.                                           *
69                              ;*                                                                             *
70                              ;* Entry:  None                                                                *
71                              ;* Exit:   SCC Channel C initialized, transmitter and receiver interrupts enabled  *
72                              ;* Alters: AL, flags                                                           *
73                              ;* Calls:  None                                                                *
74                              ;* Stack:  5 words                                                             *
75                              ;*******************************************************************************
76                              ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
77      0003                    INIT_SCC_C      PROC    FAR
78      0003  51                               PUSH    CX              ;Save registers
79      0004  52                               PUSH    DX
80      0005  56                               PUSH    SI
81      0006  EB 2C 90                         JMP     INIT_C          ;Jump past the table
82      0009  2C                SDLC_INIT_CH_C  DB      OFFSET SDLC_INIT_CH_C_END - OFFSET SDLC_INIT_CH_C-1
83      000A  09 80                             DB      09,080H         ;Register 9, Reset CH. C
84      000C  04 20                             DB      04,020H         ;Register 4, x1 Clk/SDLC
85      000E  02 F0                             DB      02,INTVECT      ;Register 2, Interrupt Vector
86      0010  03 CC                             DB      03,0CCH         ;Register  3, Rcv 8 Bits,Addr search
87      0012  05 60                             DB      05,060H         ;Register  5, Xmt 8 Bits, SDLC CRC
88      0014  06 CC                             DB      06,0CCH         ;Register  6, SDLC Address (user-selected)
89      0016  07 7E                             DB      07,07EH         ;Register  7, SDLC flag
90      0018  09 01                             DB      09,001H         ;Register  9, Vector Includes Status Low
91      001A  0A 80                             DB      10,080H         ;Register 10, Misc Control
92      001C  0B 16                             DB      11,016H         ;Register 11, Rcv Clk=RClk pin, Xmt Clk=BR Gen
93      001E  0C 7E                             DB      12,07EH         ;Register 12, Low Time Const. Byte (19.2K)
94      0020  0D 00                             DB      13,000H         ;Register 13, High Time Const. Byte (19.2K)
95      0022  0E 02                             DB      14,002H         ;Register 14, Baud Rate Gen=PCLK Input
96      0024  03 C1                             DB      03,0C1H         ;Register  3, Rcv 8 Bits, Rcv Enabled
97      0026  05 69                             DB      05,069H         ;Register  5, Xmt 8 Bits, En Xmt
98      0028  00 80                             DB      00,080H         ;Reset Xmt CRC generator
99      002A  0E 03                             DB      14,003H         ;Register 14, BR Gen=PCLK input, En BR Gen
100     002C  0F 40                             DB      15,040H         ;Register 15, Enable XMT undrun/EOM interrupts
101     002E  00 10                             DB      00,010H         ;Reset Ext/stat interrupts twice
102     0030  00 10                             DB      00,010H
103     0032  01 12                             DB      01,012H         ;Enable  Rcv and Xmt interrupts
104     0034                    SDLC_INIT_CH_C_END      LABEL BYTE      ;End of table
105
106
```

```
0034  BE 0009 R     INIT_C:     MOV     SI,OFFSET SDLC_INIT_CH_C    ;Get table address
0037  2E: AC                    LODS    SDLC_INIT_CH_C              ;Byte count to CX
0039  8A C8                    MOV     CL,AL
003B  32 ED                    XOR     CH,CH
003D  BA 00B5                  MOV     DX,SCC_CMD_C                ;Port addr to DX
0040                           INIT_C_LP:
0040  2E: AC                    LODS    SDLC_INIT_CH_C              ;Get byte
0042  EE                      OUT     DX,AL                       ;Output to SCC CH C
0043  E2 FB                    LOOP    INIT_C_LP                   ;Repeat
0045  5E                      POP     SI                          ;Restore registers
0046  5A                      POP     DX
0047  59                      POP     CX
0048  CB                      RET
0049                           INIT_SCC_C  ENDP

                               ;*****************************************************************************
                               ;* INIT_SCC_D                                                                *
                               ;*       This subroutine initializes Channel D of the SCC with the data in the table  *
                               ;*       starting at SDLC_INIT_CH_D.                                         *
                               ;*                                                                           *
                               ;* Entry:  None                                                              *
                               ;* Exit:   SCC channel D initialized, transmitter and receiver interrupts enabled  *
                               ;* Alters: AL, Flags                                                         *
                               ;* Calls:  None                                                              *
                               ;* Stack:  5 words                                                           *
                               ;*****************************************************************************
                               ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0049                           INIT_SCC_D  PROC    FAR
0049  51                      PUSH    CX              ;Save registers
004A  52                      PUSH    DX
004B  56                      PUSH    SI
004C  EB 26 90                JMP     INIT_D          ;Jump past the table
004F  26                      SDLC_INIT_CH_D  DB      OFFSET SDLC_INIT_CH_D_END - OFFSET SDLC_INIT_CH_D-1
0050  09 40                   DB      09,040H         ;Register 9, Reset CH. D
0052  04 20                   DB      04,020H         ;Register 4, x1 Clk/SDLC
0054  03 CC                   DB      03,0CCH         ;Register  3, Rcv 8 Bits, Addr search
0056  05 60                   DB      05,060H         ;Register  5, Xmt 8 Bits, SDLC CRC
0058  06 DD                   DB      06,0DDH         ;Register  6, SDLC Address (user-selected)
005A  07 7E                   DB      07,07EH         ;Register  7, SDLC flag
005C  0A 80                   DB      10,080H         ;Register 10, Misc Control
005E  0B 16                   DB      11,016H         ;Register 11, Rcv Clk=RClk pin, Xmt Clk=BR Gen
0060  0C 7E                   DB      12,07EH         ;Register 12, Low Time Const. Byte (19.2K)
0062  0D 00                   DB      13,000H         ;Register 13, High Time Const. Byte (19.2K)
0064  0E 02                   DB      14,002H         ;Register 14, Baud Rate Gen=PCLK Input
0066  03 C1                   DB      03,0C1H         ;Register  3, Rcv 8 Bits, Rcv Enabled
0068  05 69                   DB      05,069H         ;Register  5, Xmt 8 Bits, En Xmt
006A  00 80                   DB      00,080H         ;Reset Xmt CRC
006C  0E 03                   DB      14,003H         ;Register 14, BR Gen=PCLK Input, En BR Gen
006E  0F 40                   DB      15,040H         ;Register 15, Disable External/Status Interrupts
0070  01 12                   DB      01,012H         ;En Rcv and Xmt Interrupts
0072  09 09                   DB      09,09H          ;Master Int Enable
0074                           SDLC_INIT_CH_D_END  LABEL BYTE      ;End Of Table
```

```
0074  BE 004F R        INIT_D:     MOV   SI,OFFSET SDLC_INIT_CH_D    ;Get table address
0077  2E: AC                       LODS  SDLC_INIT_CH_D              ;Byte count to CX
0079  8A C8                       MOV   CL,AL
007B  32 ED                       XOR   CH,CH
007D  BA 00B7                     MOV   DX,SCC_CMD_D                ;Port add to DX
0080                    INIT_D_LP:
0080  2E: AC                       LODS  SDLC_INIT_CH_D              ;Get byte
0082  EE                         OUT   DX,AL                       ;Output to SCC CH D
0083  E2 FB                       LOOP  INIT_D_LP                   ;Repeat
0085  5E                         POP   SI            ;Restore registers
0086  5A                         POP   DX
0087  59                         POP   CX
0088  CB                         RET
0089                    INIT_SCC_D  ENDP
                        ;*****************************************************************************
                        ;* XMT_CH_C_SDLC                                                             *
                        ;*       This subroutine transmits a block of data using the SDLC format.  The program *
                        ;*       first waits for any previous transmission to be completed.  The transmit      *
                        ;*       CRC generator is reset, the secondary address is sent, the reset overrun/end  *
                        ;*       of message latch is reset, and the program exits.  Thereafter, whenever the   *
                        ;*       transmitter indicates it is ready, via interrupt, the transmit interrupt      *
                        ;*       service routine sends another byte.  When the transmitter interrupts after    *
                        ;*       sending the last byte, the transmitter interrupt is reset and no more data is *
                        ;*       sent.                                                                         *
                        ;*                                                                                     *
                        ;* Entry:  CX = Number of bytes to be sent                                             *
                        ;*         AL = SDLC secondary address                                                 *
                        ;*         XMT_BUF_C  =  Data to be sent                                               *
                        ;* Exit:   Byte count stored at (XMT_COUNT_C)                                          *
                        ;*         Data pointer stored at (XMT_BUF_C)                                          *
                        ;*         SDLC address sent                                                           *
                        ;* Alters: AL, DI, DX, SI, Flags                                                       *
                        ;* Calls:  None                                                                        *
                        ;* Stack:  3 words                                                                     *
                        ;*****************************************************************************
                        ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0089                    XMT_CH_C_SDLC   PROC    FAR
0089  50                         PUSH  AX
008A  BA 00B5                     MOV   DX,SCC_CMD_C          ;Point to control reg CH C
008D  EC                  BUFMPTY_C:    IN    AL,DX           ;Wait for transmit buffer to empty
008E  24 04                       AND   AL,04H
0090  3C 04                       CMP   AL,04H
0092  75 F9                       JNE   BUFMPTY_C
0094  B0 80                       MOV   AL,080H               ;Reset Xmt CRC Generator
0096  EE                         OUT   DX,AL
0097  BA 00B4                     MOV   DX,SCC_DTA_C          ;Point to SCC data port
009A  58                         POP   AX                    ;Send SDLC address
009B  EE                         OUT   DX,AL
009C  BA 00B5                     MOV   DX,SCC_CMD_C          ;Point to SCC control port
009F  B0 C0                       MOV   AL,0C0H               ;Reset Xmt Ovrn/EOM latch
00A1  EE                         OUT   DX,AL
00A2  CB                         RET
00A3                    XMT_CH_C_SDLC   ENDP
```

```
                        ;*****************************************************************************
                        ;* XMT_C_INT                                                                 *
                        ;*       This is the Transmit Buffer Empty interrupt service routine for Channel C.  *
                        ;*       The next character of the message is moved to the transmit buffer,  *
                        ;*       and the byte count and buffer address are adjusted.  If the last character  *
                        ;*       has been sent, the transmitter interrupt is reset.                  *
                        ;*                                                                           *
                        ;* Entry:  None                                                              *
                        ;* Exit:   Byte count decremented and stored at (XMT_COUNT_C)                *
                        ;*       Data pointer incremented and stored at (XMT_BUF_C)                  *
                        ;*       Next byte sent or transmit interrupt reset.                         *
                        ;* Alters: None                                                              *
                        ;* Calls:  None                                                              *
                        ;* Stack:  8  words                                                          *
                        ;*****************************************************************************
                        ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
00A3                    XMT_C_INT       PROC    FAR                     ;Channel C XMT Interrupt Service Routine
00A3  50                                PUSH    AX                      ;Save registers
00A4  1E                                PUSH    DS
00A5  56                                PUSH    SI
00A6  52                                PUSH    DX
00A7  06                                PUSH    ES
00A8  FB                                STI                             ;Re-enable interrupts
00A9  B8  ---- R                        MOV     AX,SEG MAILBOX
00AC  8E D8                             MOV     DS,AX
                                        ASSUME  DS:MAILBOX
00AE  83 3E 0000 R 00                   CMP     XMT_COUNT_C,0           ;Last char sent?
00B3  75 09                             JNE     CONTINUE                ;Yes, jump to end
00B5  BA 00B5                           MOV     DX,SCC_CMD_C            ;Point DX to SCC control port
00B8  B0 28                             MOV     AL,28H                  ;Reset XMT interrupt pending
00BA  EE                                OUT     DX,AL
00BB  EB 1A 90                          JMP     LAST_SDLC               ;Go to the end of the program.
00BE  FF 0E 0000 R              CONTINUE:       DEC     XMT_COUNT_C     ;No, decrement byte count
00C2  C4 36 0002 R                      LES     SI,XMT_BUF_C            ;Point to transmit buffer
                                        ASSUME  ES:BUFFER
00C6  FC                                CLD
00C7  26: AC                            LODS    XMT_BUFFER
00C9  BA 00B4                           MOV     DX,SCC_DTA_C
00CC  89 36 0002 R                      MOV     WORD PTR XMT_BUF_C,SI
00D0  EE                                OUT     DX,AL                   ;Output byte
00D1  BA 00B5                           MOV     DX,SCC_CMD_C
00D4  B0 38                             MOV     AL,038H                 ;Reset highest IUS
00D6  EE                                OUT     DX,AL
00D7  07                        LAST_SDLC:      POP     ES              ;Restore registers
00D8  5A                                POP     DX
00D9  5E                                POP     SI
00DA  1F                                POP     DS
00DB  58                                POP     AX
00DC  CF                                IRET                            ;Return
00DD                            XMT_C_INT       ENDP
```

```
                        ;*****************************************************************************
                        ;* EXT_STAT_C, EXT_STAT_D (User supplied)                                    *
                        ;*        These interrupt service routines should process External/Status interrupts
                        ;*        from Channels C and D.                                             *
                        ;*****************************************************************************
                        ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
00DD                    EXT_STAT_C      PROC FAR
00DD  CF                                IRET
00DE                    EXT_STAT_C      ENDP
00DE                    EXT_STAT_D      PROC FAR
00DE  CF                                IRET
00DF                    EXT_STAT_D      ENDP
                        ;*****************************************************************************
                        ;* RCV_CH_D_SDLC                                                             *
                        ;*        This is an interrupt service routine that receives and stores synchronously  *
                        ;*        transmitted SDLC characters.  If errors are detected, or the final flag      *
                        ;*        received, a different interrupt service routine (SPCL_RCV_D) is called.  The *
                        ;*        user must supply routines to process the received SDLC frame.                *
                        ;*                                                                                     *
                        ;* Entry:  None                                                                        *
                        ;* Exit:   Character stored in buffer, next storage address at RCV_BUF_D.              *
                        ;* Alters: None                                                                        *
                        ;* Calls:  None                                                                        *
                        ;* Stack:  8 words                                                                     *
                        ;*****************************************************************************
                        ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
00DF                    RCV_CH_D_SDLC   PROC    FAR
00DF  50                                PUSH    AX                      ;Save registers
00E0  52                                PUSH    DX
00E1  57                                PUSH    DI
00E2  1E                                PUSH    DS
00E3  06                                PUSH    ES
00E4  FB                                STI                             ;Re-enable interrupts
00E5  B8  ---- R                        MOV     AX,SEG MAILBOX
00E8  8E D8                             MOV     DS,AX
                                        ASSUME  DS:MAILBOX
00EA  BA 00B6                           MOV     DX,SCC_DTA_D            ;Point to CH D data port
00ED  C4 3E 0006 R                      LES     DI,RCV_BUF_D            ;Point ES:DI to CH D Rcv Buffer
                                        ASSUME  ES:BUFFER
00F1  EC                                IN      AL,DX                   ;Input data
00F2  FC                                CLD
00F3  AA                                STOS    RCV_BUFFER              ;Store received data in buffer
00F4  89 3E 0006 R                      MOV     RCV_BUF_D,DI            ;Store next buffer location
00F8  BA 00B7                           MOV     DX,SCC_CMD_D            ;Reset highest IUS
00FB  B0 38                             MOV     AL,038H
00FD  EE                                OUT     DX,AL
00FE  07                                POP     ES                      ;Restore registers
00FF  1F                                POP     DS
0100  5F                                POP     DI
0101  5A                                POP     DX
0102  58                                POP     AX
0103  CF                                IRET                            ;Return
0104                    RCV_CH_D_SDLC   ENDP
```

```
                                ;*****************************************************************************
                                ;* END_OF_FRAME_D, OVRN_ERR_D, CRC_ERR_D,  (All user-supplied)               *
                                ;*        These subroutines are called by the SPCL_RCV_D interrupt           *
                                ;*        service routines in response to various error conditions, and to an SDLC  *
                                ;*        end-of-frame.                                                      *
                                ;*****************************************************************************
                                ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0104                            END_OF_FRAME_D  PROC    FAR
0104  CB                                RET
0105                            END_OF_FRAME_D  ENDP

0105                            OVRN_ERR_D      PROC    FAR
0105  CB                                RET
0106                            OVRN_ERR_D      ENDP

0106                            CRC_ERR_D       PROC    FAR
0106  CB                                RET
0107                            CRC_ERR_D       ENDP


                                ;*****************************************************************************
                                ;* SPCL_RCV_D                                                                *
                                ;*        This is an interrupt service routine that indicates an SDLC CRC or receiver  *
                                ;*        overrun error, or end of frame.  The user must supply subroutines to process  *
                                ;*        the errors and interpret the data received.  After processing, the error code *
                                ;*        is reset.                                                          *
                                ;*                                                                           *
                                ;* Entry:  None                                                              *
                                ;* Exit:   Channel D RR1 status stored at (ERR_STAT_D)                       *
                                ;* Alters: None                                                              *
                                ;* Calls:  END_OF_FRAME_D, CRC_ERR_D, OVRN_ERR_D   (user-supplied)           *
                                ;* Stack:  8 words                                                           *
                                ;*****************************************************************************
                                ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0107                            SPCL_RCV_D      PROC    FAR
0107  50                                PUSH    AX                      ;Save registers
0108  52                                PUSH    DX
0109  1E                                PUSH    DS
010A  FB                                STI                             ;Re-enable interrupts
010B  B8  ---- R                        MOV     AX,SEG MAILBOX
010E  8E D8                             MOV     DS,AX
                                        ASSUME  DS:MAILBOX
0110  BA 00B7                           MOV     DX,SCC_CMD_D            ;Point to SCC CH D, RR1
0113  B0 01                             MOV     AL,001H
0115  EE                                OUT     DX,AL
0116  EC                                IN      AL,DX                   ;Read error status
0117  A2 000A R                         MOV     ERR_STAT_D,AL           ;Store it
011A  24 40                             AND     AL,040H                 ;CRC error ?
011C  74 05                             JZ      GO_ON3
011E  9A 0106 ---- R                    CALL    CRC_ERR_D               ;Yes
0123  A0 000A R                 GO_ON3: MOV     AL,ERR_STAT_D           ;Overrun error ?
0126  24 20                             AND     AL,020H
```

```
0128  74 05                              JZ      GO_ON4
012A  9A 0105 ---- R                     CALL    OVRN_ERR_D              ;Yes
012F  A0 000A R                GO_ON4:   MOV     AL,ERR_STAT_D           ;End of frame ?
0132  24 80                              AND     AL,080H
0134  74 05                              JZ      GO_ON5
0136  9A 0104 ---- R                     CALL    END_OF_FRAME_D          ;Yes
013B  B0 30                    GO_ON5:   MOV     AL,030H                 ;Error reset
013D  EE                                 OUT     DX,AL
013E  B0 38                              MOV     AL,038H                 ;Reset highest IUS
0140  EE                                 OUT     DX,AL
0141  1F                                 POP     DS                      ;Restore and return
0142  5A                                 POP     DX
0143  58                                 POP     AX
0144  CF                                 IRET
0145                           SPCL_RCV_D ENDP

                               ;*****************************************************************************
                               ;* INIT_SDLC_COMM                                                            *
                               ;*        This subroutine initializes SCC Channels C and D in synchronous SDLC format  *
                               ;*        and enables interrupts.  The initialized conditions should be checked       *
                               ;*        to be sure they match the application.  The baud rate is 19.2K for          *
                               ;*        both channels.                                                              *
                               ;*                                                                                    *
                               ;* Entry:  None                                                                       *
                               ;* Exit:   SCC initialized, vector table in RAM, interrupts enabled                   *
                               ;* Alters: AX, DX, DI, Flags                                                          *
                               ;* Calls:  INIT_SCC_C, INIT_SCC_D                                                     *
                               ;* Stack:  4 words                                                                    *
                               ;*****************************************************************************
                               ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0145                           INIT_SDLC_COMM  PROC    FAR
0145  9A 0003 ---- R                        CALL    INIT_SCC_C
014A  9A 0049 ---- R                        CALL    INIT_SCC_D
014F  CB                                    RET
0150                           INIT_SDLC_COMM  ENDP
```

```
425
426                             ;*****************************************************************************
427                             ;* INIT_VECTORS                                                              *
428                             ;*   This subroutine initializes the vector table for vectors produced by the SCC.  *
429                             ;*                                                                           *
430                             ;* Entry:  None                                                              *
431                             ;* Exit:   Vector table initialized and stored in RAM.                       *
432                             ;* Alters: None                                                              *
433                             ;* Calls:  None                                                              *
434                             ;* Stack:  4 words                                                           *
435                             ;*****************************************************************************
436                             ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
437     0150                    INIT_VECTORS    PROC    FAR
438     0150  50                               PUSH    AX                      ;Save registers
439     0151  1E                               PUSH    DS
440     0152  B8  ---- R                       MOV     AX,SEG VECTOR
441     0155  8E D8                            MOV     DS,AX
442                                            ASSUME  DS:VECTOR
443     0157  C7 06 03C8 R 00DE R              MOV     WORD PTR INTR1,OFFSET EXT_STAT_D
444     015D  C7 06 03CA R  ---- R             MOV     WORD PTR INTR1+2,SEG EXT_STAT_D
445     0163  C7 06 03D0 R 00DF R              MOV     WORD PTR INTR2,OFFSET RCV_CH_D_SDLC
446     0169  C7 06 03D2 R  ---- R             MOV     WORD PTR INTR2+2,SEG RCV_CH_D_SDLC
447     016F  C7 06 03D8 R 0107 R              MOV     WORD PTR INTR3,OFFSET SPCL_RCV_D
448     0175  C7 06 03DA R  ---- R             MOV     WORD PTR INTR3+2,SEG SPCL_RCV_D
449     017B  C7 06 03E0 R 00A3 R              MOV     WORD    PTR     INTR4,OFFSET    XMT_C_INT
450     0181  C7 06 03E2 R  ---- R             MOV     WORD    PTR     INTR4+2,SEG     XMT_C_INT
451     0187  C7 06 03E8 R 00DD R              MOV     WORD    PTR     INTR5,OFFSET    EXT_STAT_C
452     018D  C7 06 03EA R  ---- R             MOV     WORD    PTR     INTR5+2,SEG     EXT_STAT_C
453     0193  1F                               POP     DS                      ;Restore registers
454     0194  58                               POP     AX
455     0195  CB                               RET                     ;Return
456     0196                    INIT_VECTORS    ENDP
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
```

```
                        ;*****************************************************************************
                        ;*SDLC_COMM                                                                  *
                        ;*   This is the main routine that initializes channels C and D of the SCC  *
                        ;*   initializes the vector table, transmits the first character (SDLC address),  *
                        ;*   then waits for the SCC to interrupt.  When the last character has been  *
                        ;*   transmitted, the routine exits to LAST_SDLC.                           *
                        ;*                                                                          *
                        ;* Entry:  None                                                            *
                        ;* Exit:   Vector table initialized and stored in RAM.                     *
                        ;* Alters: None                                                            *
                        ;* Calls:  None                                                            *
                        ;* Stack:  4 words                                                         *
                        ;*****************************************************************************
                        ASSUME  CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0196                    SDLC_COMM       PROC    NEAR
0196  9A 0145 ---- R                   CALL    INIT_SDLC_COMM          ;Initialize SCC channels C and D
019B  9A 0150 ---- R                   CALL    INIT_VECTORS            ;Initialize vector table
01A0  FB                               STI                             ;Enable interrupts
01A1  B8  ---- R                       MOV     AX,SEG MAILBOX
01A4  8E D8                            MOV     DS,AX
                                       ASSUME  DS:MAILBOX
01A6  C7 06 0000 R 000F                MOV     XMT_COUNT_C,0FH
01AC  C7 06 0002 R 0400                MOV     WORD PTR XMT_BUF_C,0400H        ;Point to transmit buffer address
01B2  C7 06 0004 R 0100                MOV     WORD PTR XMT_BUF_C+2,0100H
01B8  C7 06 0006 R 0800                MOV     WORD PTR RCV_BUF_D,0800H        ;Point to receive buffer address
01BE  C7 06 0008 R 0100                MOV     WORD PTR RCV_BUF_D+2,0100H
01C4  B0 DD                            MOV     AL,0DDH                 ;Select SDLC secondary address
01C6  9A 0089 ---- R                   CALL    XMT_CH_C_SDLC
01CB  83 3E 0000 R 00   WAIT_LOOP:     CMP     XMT_COUNT_C,0           ;End of transmission?
01D0  75 F9                            JNE     WAIT_LOOP
01D2  EB FE             END_SDLC:      JMP     END_SDLC
01D4                    SDLC_COMM       ENDP
01D4                    CODE    ENDS
                        END
```

Segments and groups:

N a m e	Size	align	combine	class
BUFFER	0002	AT	0000	
CODE	01D4	PARA	PUBLIC	
MAILBOX	000B	AT	0100	
VECTOR	0400	AT	0000	

Symbols:

N a m e	Type	Value	Attr	
BUFMPTY_C	L NEAR	008D	CODE	
CONTINUE	L NEAR	00BE	CODE	
CRC_ERR_D	F PROC	0106	CODE	Length =0001
END_OF_FRAME_D	F PROC	0104	CODE	Length =0001
END_SDLC	L NEAR	01D2	CODE	
ERR_STAT_D	L BYTE	000A	MAILBOX	
EXT_STAT_C	F PROC	00DD	CODE	Length =0001
EXT_STAT_D	F PROC	00DE	CODE	Length =0001
GO_ON3	L NEAR	0123	CODE	
GO_ON4	L NEAR	012F	CODE	
GO_ON5	L NEAR	013B	CODE	
INIT_C	L NEAR	0034	CODE	
INIT_C_LP	L NEAR	0040	CODE	
INIT_D	L NEAR	0074	CODE	
INIT_D_LP	L NEAR	0080	CODE	
INIT_SCC_C	F PROC	0003	CODE	Length =0046
INIT_SCC_D	F PROC	0049	CODE	Length =0040
INIT_SDLC_COMM	F PROC	0145	CODE	Length =000B
INIT_VECTORS	F PROC	0150	CODE	Length =0046
INTR0	L DWORD	03C0	VECTOR	
INTR1	L DWORD	03C8	VECTOR	
INTR2	L DWORD	03D0	VECTOR	
INTR3	L DWORD	03D8	VECTOR	
INTR4	L DWORD	03E0	VECTOR	
INTR5	L DWORD	03E8	VECTOR	
INTR6	L DWORD	03F0	VECTOR	
INTR7	L DWORD	03F8	VECTOR	
INTRX0	L DWORD	03C4	VECTOR	
INTRX1	L DWORD	03CC	VECTOR	
INTRX2	L DWORD	03D4	VECTOR	
INTRX3	L DWORD	03DC	VECTOR	
INTRX4	L DWORD	03E4	VECTOR	
INTRX5	L DWORD	03EC	VECTOR	
INTRX6	L DWORD	03F4	VECTOR	
INTRX7	L DWORD	03FC	VECTOR	
INTVECT	Number	00F0		
LAST_SDLC	L NEAR	00D7	CODE	
OVRN_ERR_D	F PROC	0105	CODE	Length =0001
RCV_BUFFER	L BYTE	0001	BUFFER	
RCV_BUF_D	L DWORD	0006	MAILBOX	

```
RCV_CH_D_SDLC. . . . . .      F PROC  00DF    CODE    Length =0025
SCC_CMD_C. . . . . .         Number  00B5
SCC_CMD_D. . . . . .         Number  00B7
SCC_DTA_C. . . . . .         Number  00B4
SCC_DTA_D. . . . . .         Number  00B6
SDLC_COMM. . . . . .         N PROC  0196    CODE    Length =003E
SDLC_INIT_CH_C . . . . . .   L BYTE  0009    CODE
SDLC_INIT_CH_C_END . . . . . .  L BYTE  0034    CODE
SDLC_INIT_CH_D . . . . . .   L BYTE  004F    CODE
SDLC_INIT_CH_D_END . . . . . .  L BYTE  0074    CODE
SPCL_RCV_D . . . . . .       F PROC  0107    CODE    Length =003E
TEST_SDLC. . . . . .         F PROC  0000    CODE    Length =0003
WAIT_LOOP. . . . . .         L NEAR  01CB    CODE
XMT_BUFFER . . . . . .       L BYTE  0000    BUFFER
XMT_BUF_C. . . . . .         L DWORD 0002    MAILBOX
XMT_CH_C_SDLC. . . . . .     F PROC  0089    CODE    Length =001A
XMT_COUNT_C. . . . . .       L WORD  0000    MAILBOX
XMT_C_INT. . . . . .         F PROC  00A3    CODE    Length =003A

Warning Severe
Errors  Errors
0       0
```

Section 5
MAINTENANCE

REFERENCE DRAWINGS

The schematic (Figure 5-1) and assembly drawing (Figure 5-2) in the following pages are included in this manual FOR REFERENCE USE ONLY. These drawings may differ in some respects from the card you receive from Pro-Log. If you require the specific revision documents of the schematic and assembly for the board shipped, please call Pro-Log Customer Support at (800) 538-9570, extension 3617. In California call (408) 646-3617 from 7 a.m. to 4 p.m. (Pacific Time).

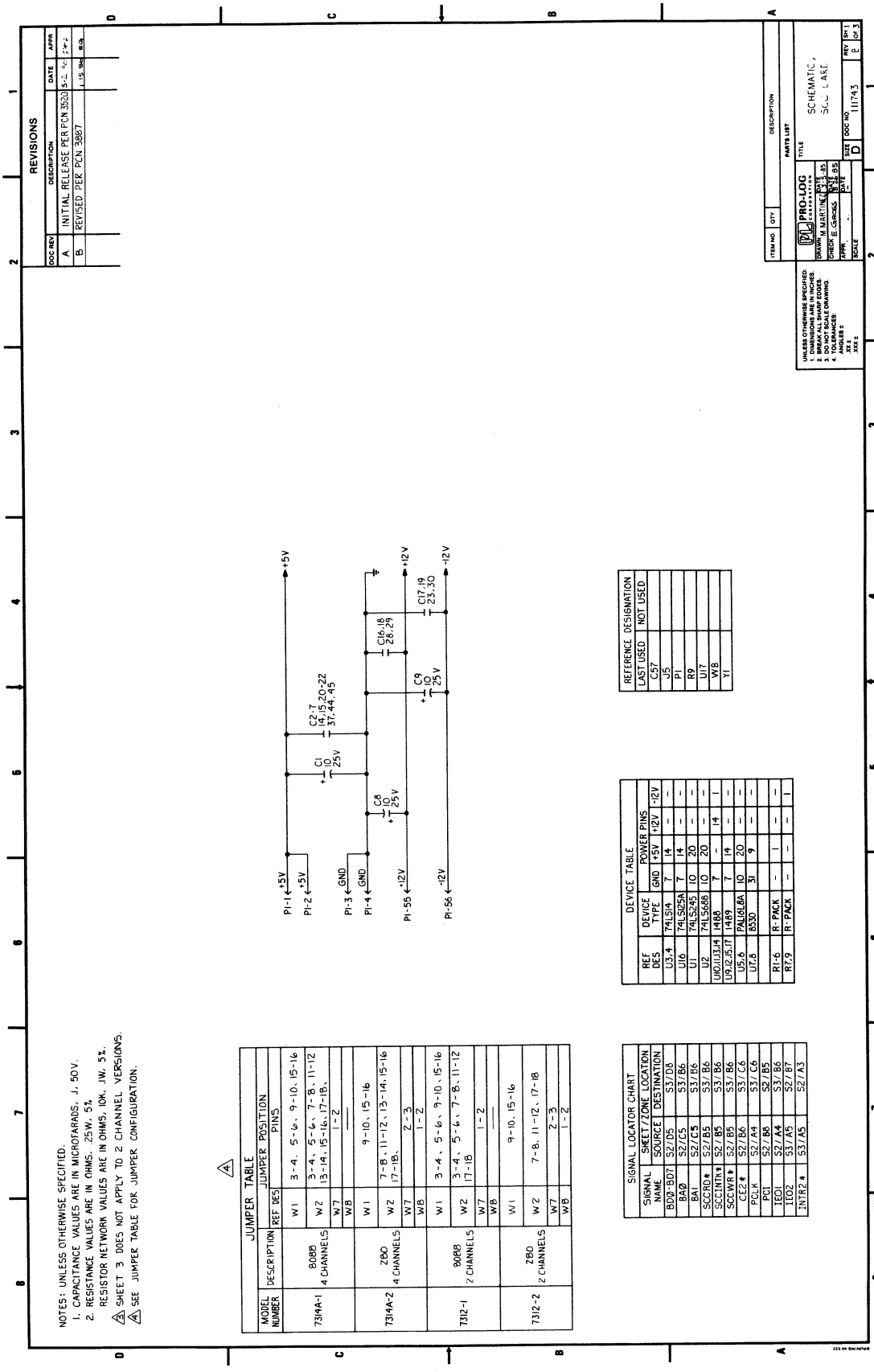


Figure 5-1. Schematic (Page 1 of 3)
FOR REFERENCE ONLY

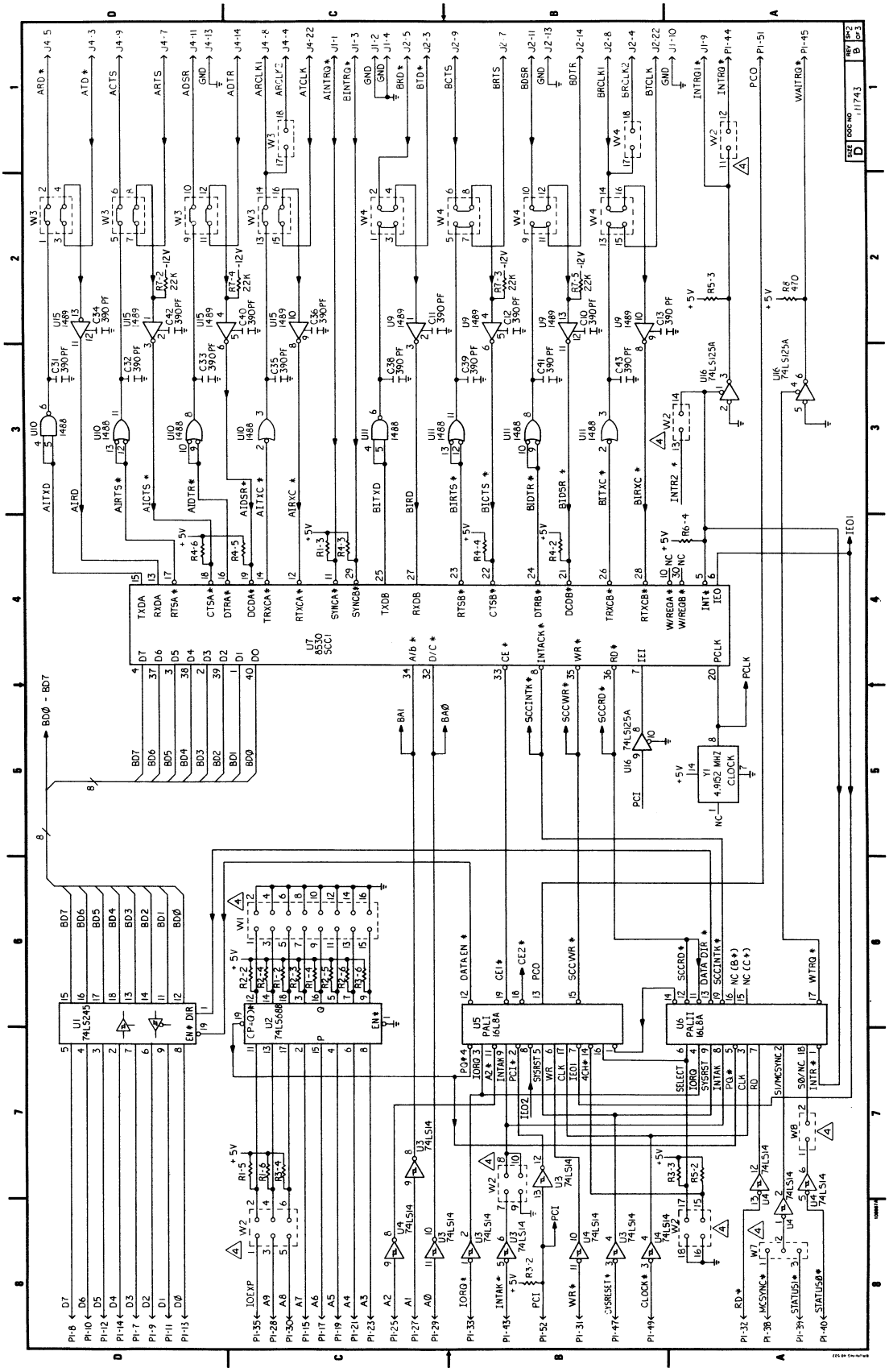


Figure 5-1. Schematic (Page 2 of 3)
FOR REFERENCE ONLY

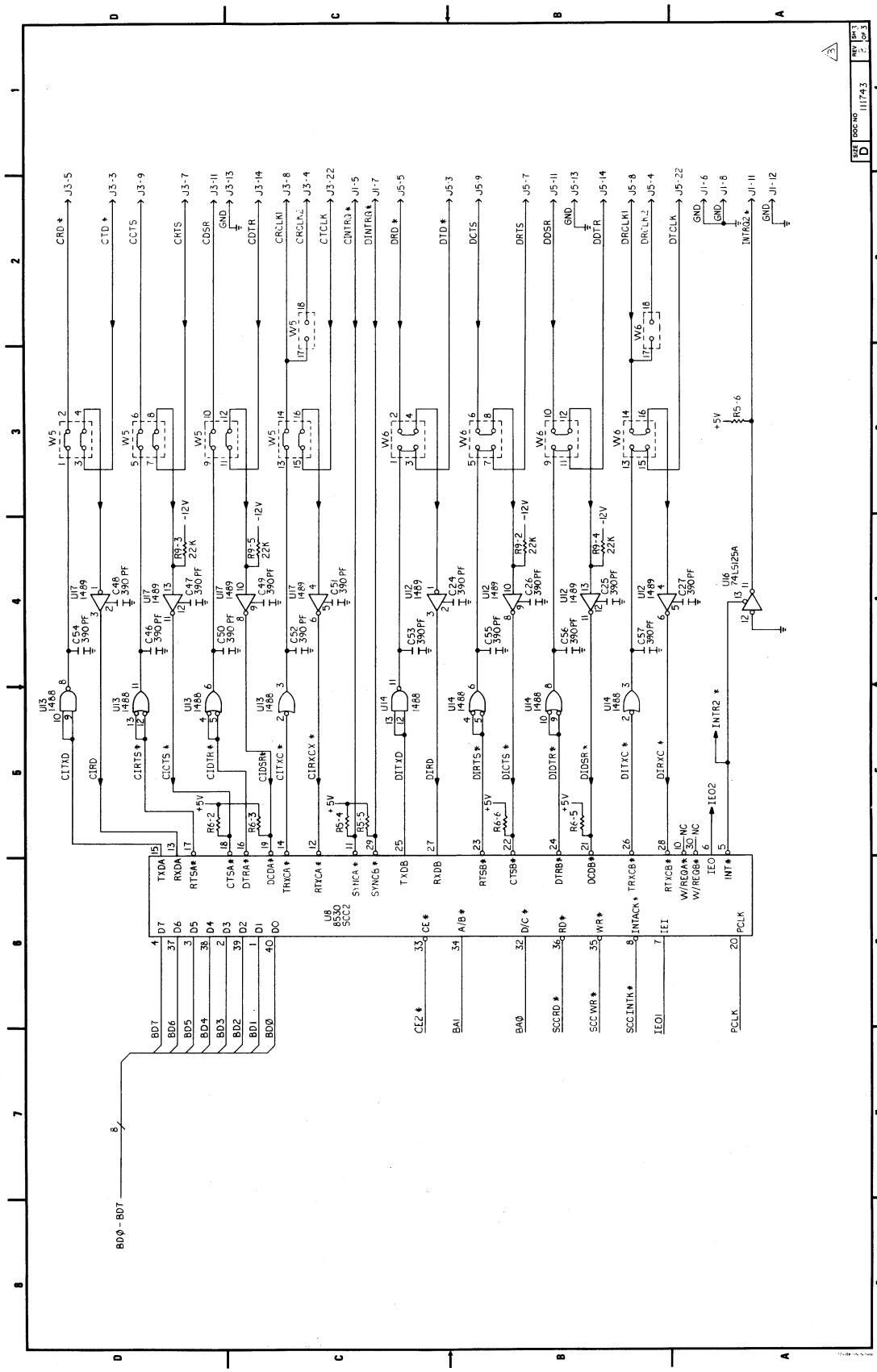


Figure 5-1. Schematic (Page 3 of 3)
FOR REFERENCE ONLY

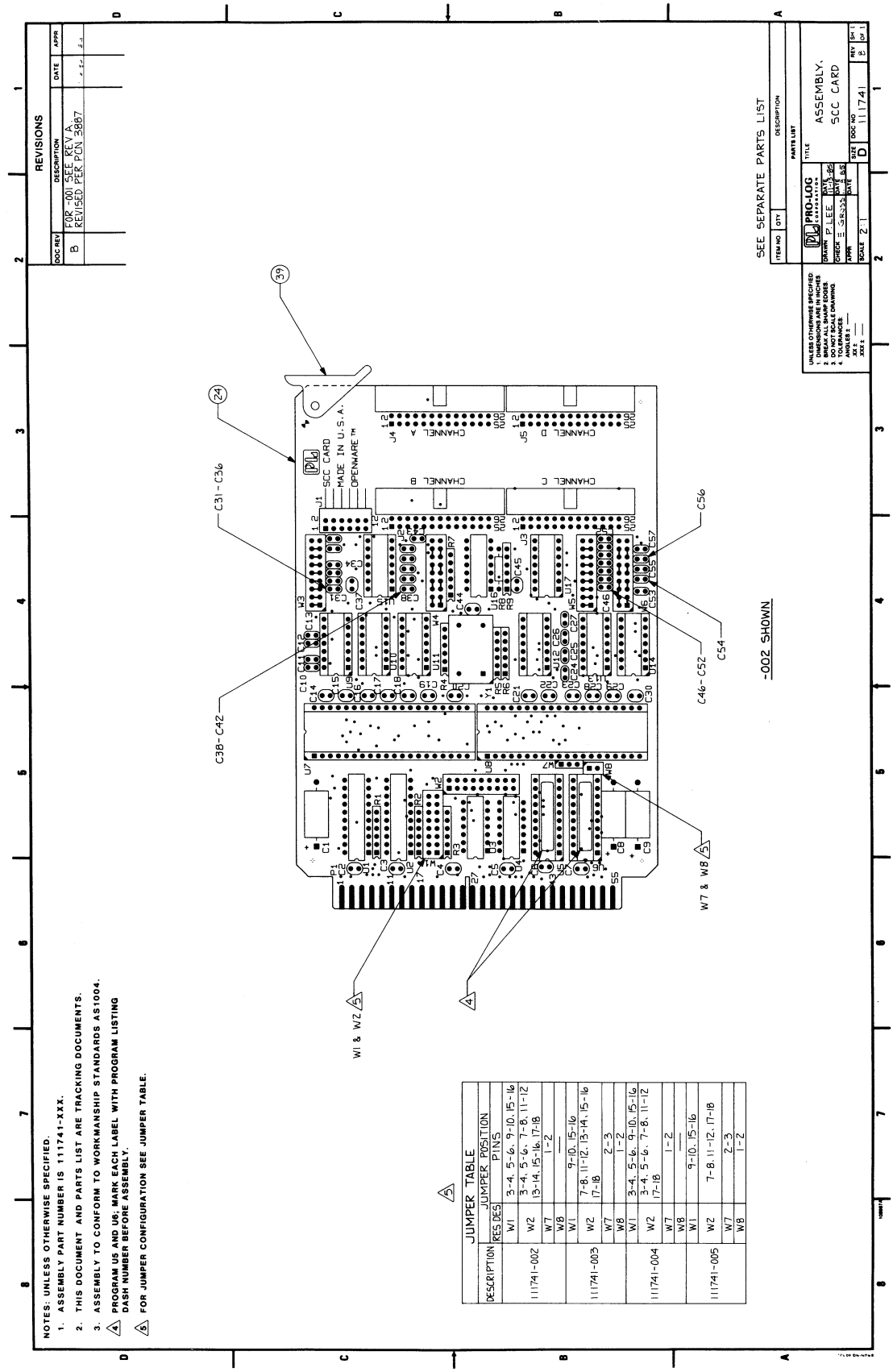


Figure 5-2. Assembly Drawing
FOR REFERENCE ONLY

RETURN FOR REPAIR PROCEDURES

Domestic Customers:

1. Call Customer Support at (800) 538-9570, ext. 3617. In California call (408) 646-3617 from 7 a.m. to 4 p.m. (Pacific Time).
2. Explain the problem and we may be able to solve it on the phone. If not, we will give you a Customer Return Order (CRO) number.

   Mark the CRO number on the shipping label, packing slip, and other paperwork accompanying the return. We cannot accept returns without a CRO.
3. Please be sure to enclose a packing slip with CRO number, serial number of the equipment, if applicable, and reason for return, with the name and telephone number of the person we should contact (preferably the user), if we have any further questions.
4. Pack the equipment in a solid cardboard box secured with packing material.

CAUTION

Loose MOS integrated circuits, or any product containing CMOS integrated circuits, must be protected from electrostatic discharge during handling. Use conductive foam pads or conductive plastic bags, and never place MOS or CMOS circuitry in contact with Styrofoam materials.

5. Ship prepaid and insured to:

   Pro-Log Corporation
   2560 Garden Road
   Monterey, California 93940

   Reference CRO #_____

International Customers:

Equipment repair is handled by your local Pro-Log distributor. If you need to contact Pro-Log, the factory can be reached at any time by TWX at 910-360-7082.

Limited Warranty

Limited Warranty: Seller warrants that the articles furnished hereunder are free from defects in material and workmanship and perform to applicable, published Pro-Log specifications for two years from date of shipment on PROM Programmers and STD BUS Series 7000 cards. This warranty is in lieu of any other warranty expressed or implied. In no event will Seller be liable for special or consequential damages as a result of any alleged breach of this warranty provision. The liability of Seller hereunder shall be limited to replacing or repairing, at its option, any defective units which are returned F.O.B. Seller's plant. Equipment or parts which have been subject to abuse, misuse, accident, alteration, neglect, unauthorized repair or installation are not covered by warranty. Seller shall have the right of final determination as to the existence and cause of defect. As to items repaired or replaced, the warranty shall continue in effect for the remainder of the warranty period, or for ninety (90) days following date of shipment by Seller of the repaired or replaced part whichever period is longer. No liability is assumed for expendable items such as lamps and fuses. No warranty is made with respect to custom equipment or products produced to Buyer's specifications except as specifically stated in writing by Seller and contained in the contract.

Cut Here ✂

FOLD

FOLD

Cut Here ✂

PRO-LOG® CORPORATION

Yes, I'm interested in Pro-Log.

Please send me more information about your:

- ☐ 7000 Series STD BUS Cards
- ☐ STD BUS Packaging Solutions
- ☐ STD BUS Subsystem
- ☐ STD BUS Systems
- ☐ Designer Software
- ☐ Design Courses and Seminars
- ☐ Programmer Product Line
- ☐ Other _____

- ☐ I am currently using Pro-Log equipment.
- ☐ Add my name to your mailing list.
- ☐ Please have a Sales Representative contact me.

Reader Comments:

Name _____ City _____

Company _____ State _____ Zip _____

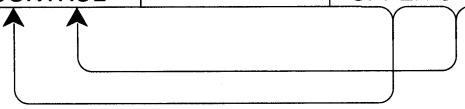
Address _____ Telephone _____

PLACE
POSTAGE
HERE

PRO-LOG®
CORPORATION
2560 GARDEN ROAD
MONTEREY, CA 93940

Attn: Literature Department

HEX				MSD	p=1	8	9	A	B	C	D	E	F
					p=0	0	1	2	3	4	5	6	7
	BITS				b8	p	p	p	p	p	p	p	p
					b7	0	0	0	0	1	1	1	1
					b6	0	0	1	1	0	0	1	1
LSD	b4	b3	b2	b5 / b1		0	1	0	1	0	1	0	1
0	0	0	0	0		NUL	DLE	SP	0	@	P	‘	p
1	0	0	0	1		SOH	DC1	!	1	A	Q	a	q
2	0	0	1	0		STX	DC2	"	2	B	R	b	r
3	0	0	1	1		ETX	DC3	#	3	C	S	c	s
4	0	1	0	0		EOT	DC4	$	4	D	T	d	t
5	0	1	0	1		ENQ	NAK	%	5	E	U	e	u
6	0	1	1	0		ACK	SYN	&	6	F	V	f	v
7	0	1	1	1		BEL	ETB	’	7	G	W	g	w
8	1	0	0	0		BS	CAN	(	8	H	X	h	x
9	1	0	0	1		HT	EM	)	9	I	Y	i	y
A	1	0	1	0		LF	SUB	*	:	J	Z	j	z
B	1	0	1	1		VT	ESC	+	;	K	[	k	{
C	1	1	0	0		FF	FS	,	<	L	\	l	|
D	1	1	0	1		CR	GS	—	=	M	]	m	}
E	1	1	1	0		SO	RS	.	>	N	^	n	~
F	1	1	1	1		SI	US	/	?	O	_	o	DEL
						CONTROL				UPPER CASE		LOWER CASE	



CONTROL CHARACTERS AND STANDARD KEYBOARD CONTROL SHIFT CHARACTERS

CTRL	@	NUL	Null	CTRL	K	VT	Vertical Tabulation	CTRL	V	SYN	Synchronous Idle
CTRL	A	SOH	Start of Heading	CTRL	L	FF	Form Feed	CTRL	W	ETB	End of Transmission Block
CTRL	B	STX	Start of Text	CTRL	M	CR	Carriage Return	CTRL	X	CAN	Cancel
CTRL	C	ETX	End of Text	CTRL	N	SO	Shift Out	CTRL	Y	EM	End of Medium
CTRL	D	EOT	End of Transmission	CTRL	O	SI	Shift In	CTRL	Z	SUB	Substitute
CTRL	E	ENQ	Enquiry	CTRL	P	DLE	Data Link Escape	CTRL	[	ESC	Escape
CTRL	F	ACK	Acknowledge	CTRL	Q	DC1	Device Control 1	CTRL	\	FS	File Separator
CTRL	G	BEL	Bell	CTRL	R	DC2	Device Control 2	CTRL	]	GS	Group Separator
CTRL	H	BS	Backspace	CTRL	S	DC3	Device Control 3	CTRL	^	RS	Record Separator
CTRL	I	HT	Horizontal Tabulation	CTRL	T	DC4	Device Control 4	CTRL	_	US	Unit Separator
CTRL	J	LF	Line Feed	CTRL	U	NAK	Negative Acknowledge			DEL	Delete

NOTE: The table above lists the keyboard character to be pressed with the Control key, the mnemonic for the function to be performed, and the description of the function. While the keyboard character is listed as uppercase, either uppercase or lowercase characters may be used.

ASCII Code Assignments

USER'S MANUAL

PRO-LOG®
CORPORATION

2560 Garden Road, Monterey, California 93940
Phone: (408) 372-4593, (800) 538-9570
Telex: 171879, TWX: 910-360-7082

112251-001A 2K 2/86